COMPUTATIONAL SEISMIC INTERPRETATION USING GEOMETRIC REPRESENTATION AND TENSOR-BASED TEXTURE ANALYSIS

A Dissertation
Presented to
The Academic Faculty

By

Zhen Wang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2018

COMPUTATIONAL SEISMIC INTERPRETATION USING GEOMETRIC REPRESENTATION AND TENSOR-BASED TEXTURE ANALYSIS

Approved by:

Dr. Ghassan AlRegib
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. James H. McClellan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Faramarz Fekri
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Zhigang Peng
School of Earth and Atmospheric Sciences
Georgia Institute of Technology

Dr. Mohamed A. Deriche
Electrical Engineering Department
King Fahd University of Petroleum & Minerals

Date Approved: May 11, 2018

We do not know a truth without knowing its cause

Aristotle

I dedicate this thesis to,

My parents, **Jianguo Wang** and **Lanting Liu**,

For their endless love, support, and encouragement, which get me where I am today.

My wife, **Yuting Hu**,

The source of happiness and strength, who gave me hope and courage to face challenges.

ACKNOWLEDGEMENTS

I am deeply indebted to my advisor, Prof. Ghassan AlRegib, for his tremendous advice, support, and help throughout this research. During my first two years of a Ph.D., Prof. AlRegib provided with me every bit of guidance and assistance I need. He taught me how to develop critical thinking skills, define research problems, conduct literature survey, make impressive presentations, and write high-quality research papers. After I was ready to conduct research by myself, he gave me enough freedom to work on topics I was interested and kept contributing valuable feedback, suggestions, and encouragement. After six years of collaborations, he is not only my PhD advisor, but also one closest friend in my life. Research is not a solo act, and collaborations accelerate the research progress and enhance the research quality. I would like to extend my sincere thanks to all former and current members of the Omni Lab for Intelligent Visual Engineering and Science (OLIVES) and the Center for Energy and Geo Processing (CEGP) for being my colleagues and friends. I am especially grateful to post-docs Zhiling Long, Haibin Di, and Can Temel as well as our current lab members Muhammed Amir Shafiq, Yazeed Alaudah, Motaz Al-Farraj, Chih-Yao Mao, Min-Hun Cheng, Tariq Alshawi, Mohit Prabhushankar, Gukyeong Kwon, Charlie Lehman, and Jinsol Lee.

I would like to thank Prof. McClellan, Prof. Fekri, Prof. Deriche, and Prof. Peng for serving on my PhD dissertation committee. Their beneficial comments and invaluable advice bring this dissertation to completion. In addition, I would like to extend my gratitude to all faculty in the Center of Signal and Information Processing (CSIP), especially Prof. Romberg, Prof. Zhou, and Prof. Ma for the substantial influence that their courses have had on my research. I also wish to thank Ms. Jane Chisholm, the instructor from the Center for Teaching and Learning (CETL), for her help on improving my academic writing and presentation skills. Without numerous instructions from her, my research papers would not have been published in good shape.

I would like to express my deepest appreciation to my family for their love and support through the Ph.D. journey. I would like to especially thank my wife, without whose adequate understanding, constant encouragement, and great sacrifice the completion of the journey would not have been possible. Many thanks go to my friends for your company in this challenging but enjoyable journey. Finally, I wish to acknowledge all who assisted me directly or indirectly in past years. Because of you, six years' stay at Georgia Tech will be the most memorable experience in my life.

TABLE OF CONTENTS

Acknowledgments v

List of Tables xi

List of Figures xii

Chapter 1: Introduction and Background 1

Chapter 2: Literature Survey 6

2.1 Computational Seismic Interpretation 6

2.1.1 Data Representation 6

2.1.2 Texture Analysis 7

2.1.3 Geological Constraint 8

2.2 Literature Survey of Computational Interpretation on Fault and Salt-dome Structures 9

2.2.1 Fault Interpretation 9

2.2.2 Salt-dome Interpretation 14

Chapter 3: The Role of Geometric Representation in Interactive Fault Interpretation 18

3.1 Hough Transform and Motion Vectors 18

3.1.1 2D Hough Transform 18

3.1.2 3D Hough Transform 20
3.1.3 Tracking Vectors 22
3.2 Interactive Fault Interpretation Framework 23
3.2.1 Preprocessing 24
3.2.2 Fault Detection in Reference Sections 26
3.2.3 Fault Tracking Through Predicted Sections 31
3.2.4 Fault Similarity Index Measurement 35
3.3 Experimental Results 39
3.3.1 Fault Detection in Reference Sections 39
3.3.2 Fault Tracking Through Predicted Sections 44
3.3.3 Overall Comparison 48
3.4 Summary 50

Chapter 4: Texture Analysis and Noise-Adjusted Tensor-based Incremental Learning in Interactive Salt-dome Interpretation 52

4.1 Texture Attributes for Salt-dome Interpretation 54
4.1.1 Gray-level Co-occurrence Matrix (GLCM) 54
4.1.2 Gradient of Texture (GoT) 56
4.2 Tensors and Multilinear Algebra 58
4.3 Interactive Salt-dome Interpretation Framework 61
4.3.1 Salt-dome Detection in Reference Sections 61
4.3.2 Salt-dome Tracking Through Predicted Sections 64
4.3.3 Similarity Index Measurement of Salt-dome Boundaries 75
4.4 Experimental Results 76

4.4.1 Salt-dome Detection 76

4.4.2 Salt-dome Tracking 84

4.5 Summary 91

Chapter 5: Tensor-based Subspace Learning and Its application in Salt-dome Tracking 93

5.1 Multilinear Dimensionality Reduction 93

5.1.1 Multilinear Dimensionality Reduction Model 93

5.1.2 Tensor Linearity Projection Preserving (TLPP) 94

5.2 Tensor Orthogonal Locality Discriminant Projection with Maximum Margin Criterion (TOLDP-MMC) 97

5.2.1 Tensor Locality-preserved Diversity Projection 98

5.2.2 Tensor Maximum Margin Criterion 100

5.2.3 Orthogonal Multilinear Transformation 102

5.3 TOLDP-MMC in Gait Recognition and Salt-dome Tracking 104

5.3.1 Gait Recognition 104

5.3.2 Salt-dome Tracking 106

5.4 Summary 114

Chapter 6: Conclusion 115

Appendix A: Derivations of total and within-class scatter matrices of tensor samples. 119

A.1 Within-class Scatter Matrix of Tensor Samples 120

A.2 Total-class Scatter Matrix of Tensor Samples 121

References 132

Vita 133

LIST OF TABLES

3.1 The FauSIM indices of faults detected by different detection methods in seismic sections 43

3.2 The comparison of FauSIM indices between tracked and detected faults in seismic sections 46

3.3 The means and standard deviations of FauSIM indices in Fig. 3.19 49

4.1 The statistical measures of SalSIM indices in Figure 4.13 obtained from various detection methods. 81

4.2 The statistical measures of the SalSIM indices of boundaries delineated by various detection methods in the crossline dataset. 82

4.3 The statistical measures of SalSIM indices in Figure 4.18. 88

5.1 The capturing conditions of training and testing data sets. 105

5.2 Compare the performance of different subspace learning methods on gait recognition. 106

5.3 The objective comparison between the salt-dome tracking method based on the TOLDP-MMC and the tracking method in Chapter 4 113

LIST OF FIGURES

1.1 Examples of raw seismic traces and a migrated seismic section. 2

2.1 (a) A normal fault near Kingman, Arizona [50], and (b) the illustration of a fault trap [51]. 9

2.2 Examples of seismic sections containing single and multiple faults. 10

2.3 (a) Jashak salt dome, known as Dashti salt dome, is located in Dashti region of Bushehr province in Southern Iran [88], and (b) the illustration of a salt trap [89]. 14

2.4 Examples of salt domes in seismic sections. 14

3.1 The 2D Hough transform represents a mapping procedure between the image space and the parameter space. 19

3.2 The normal vector in 3D spherical coordinate system. 20

3.3 The mapping relationship between volume and parameter spaces in the 3D Hough transform. 21

3.4 Motion vector v between two successive frames $t-1$ and t. 23

3.5 The interactive fault interpretation method has two main parts, fault detection in reference sections and fault tracking in predicted sections. 24

3.6 An example of a seismic section (Inline #268) and its corresponding discontinuity map. 25

3.7 (a) Binary image B containing likely fault points; (b) sinusoid curves mapped from fault points 26

3.8 The process of false feature removal: (a) types of false features, (b) a fitted fault based on the midpoints of all lines, (c) the illustration of *absolute distance*, and (d) the illustration of *lateral distance*. 27

3.9 The process of fault labeling: (a) the remaining lines after false feature removal, (b) initial labeling, (c) the searching results based on the discontinuity map, (d) optimized labeling, and (e) the labeled fault. 30

3.10 The process of tracking-vector-based projection. 32

3.11 (a) The illustration of overlaps in the crossline direction, (b) projected faults $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ and line $\mathbf{L}_{p_m}$ with the largest discontinuity values, and (c) the synthesized fault based on fault tracking. 33

3.12 (a) the comparison of an extracted fault and its ground truth labeled by interpreters, and (b) The Fréchet distance between curves A and B can be represented by the length of the magenta line. 37

3.13 The seismic section of a local volume that contains multiple faults 40

3.14 The process of fault labeling, (a) extracted fault features and the fitted line, (b) remaining features after false feature removal form the initial labeling, (c) searching results based on the discontinuity map, (d) the optimized labeling, (e) the detected fault, (f) the fault (yellow) detected by our fault detection method and the manually picked fault (green), and (g) the fault (yellow) labeled by Hale's method [66] and the manually picked fault (green). 41

3.15 The detection of multiple faults, (a) the discontinuity map of Inline #244, (b) highlighted likely fault regions, (c) detected features in different groups labeled by different colors, (d) remaining features after removing false features, (e) the initial labeling of faults, (f) detected faults with geological constraints involved, (g) the comparison between faults detected by our detection method and the manually picked fault, and (h) the comparison between faults detected by Hale's method and the manually picked fault. . . 42

3.16 The synthesis of the tracked fault in Inline #258, (a) the reference fault detected in Inline #248, (b) the reference fault detected in Inline #268, (c) the projected fault, $\mathbf{L}_{p_1}$, in solid curve, (d) the projected fault, $\mathbf{L}_{p_2}$, in solid curve, (e) $\mathbf{L}_{p_1}$, $\mathbf{L}_{p_2}$, and $\mathbf{L}_{p_m}$ for the synthesis of the tracked fault, (f) the comparison between the tracked fault (yellow) and the manually picked fault (green), and (g) the comparison between the detected fault (yellow) and the manually picked fault (green). 44

3.17 The synthesis of tracked faults in Inline #258, (a) reference faults detected in Inline #244, (b) reference faults detected in Inline #254, (c) projected faults, $\mathbf{L}_{p_1}$, in solid curves, (d) projected faults, $\mathbf{L}_{p_2}$, in solid curves, (e) the comparison between tracked faults in Inline #249 and the manually picked fault, and (d) the comparison between detected faults in Inline #249 and the manually picked fault. 47

3.18 3D fault surfaced delineated by the combined method. 49

3.19 The FauSIM indices of faults in each seismic section (From Inline #248 to Inline #286) delineated by different detection methods. 49

4.1 A seismic section contains a salt-dome, along the boundary of which local regions have different appearances. 53

4.2 The GoT value at the center point represents texture dissimilarity between two square neighboring windows. 57

4.3 The unfolding of a third-order tensor along three modes. 59

4.4 The diagram of the interactive salt-dome interpretation method. 61

4.5 Unfolding matrices of a 3-order tensor containing original and newly arrived data. In unfolding matrices along different modes, the positions of new data are different. 68

4.6 The block diagram of the adaptive classification of texture tensors. 70

4.7 (a) The priority of points in the 3×3 neighborhood of p_1 and (b) The 1-mode unfolding matrices of a tensor pair extracted from the local area of a seismic section and its corresponding GoT map. 71

4.8 The block diagram of the identification of tracked points. 73

4.9 Local comparisons between the salt-dome boundary labeled by the interactive interpretation method and the ground truth labeled manually 75

4.10 A seismic section (Inline #400) of the local volume contains the cross-section of a salt dome. 77

4.11 (a), (c), and (e) illustrate the GoT, GLCM contrast, and gradient maps of Inline #400, respectively; (b), (d), and (f) compare the manually labeled red ground truth with the green salt dome boundaries detected from attribute maps, (a), (c), and (e), respectively. 78

4.12 Local salt-dome boundaries extracted from the GoT, GLCM contrast, and gradient maps are labeled in green. The ground truth labeled manually is labeled in red. 79

4.13 The SalSIM indices of salt-dome boundaries detected from the GoT, GLCM contrast, and gradient maps. 80

4.14 The SalSIM indices of salt-dome boundaries detected from the GoT, GLCM contrast, and gradient maps. 81

4.15 The averaged SalSIM indices of salt-dome boundaries detected from noisy seismic sections using GoT- and GLCM-based detection methods. 82

4.16 (a), (b), (c), and (d) show the comparison between the red ground truth and the green tracked salt dome boundaries of Inline #391 synthesized by the tensor-based tracking method with GoT maps, the tracking method based on vectorization, the tensor-based tracking method without GoT maps, and the tensor-based tracking method with GLCM contrast maps , respectively. 86

4.17 (a), (b), (c), and (d) contain four local regions extracted from Figures 4.16(a) to (d), respectively. The red and green curves represent tracked salt-dome boundaries and the ground truth, respectively. 87

4.18 The SalSIM indices of tracked boundaries in predicted sections synthesized based on different tracking strategies. 88

4.19 (a) tracked boundary of Inline #408 and (d) detected boundary of Inline #408. 89

4.20 The SalSIM indices of tracked boundaries in predicted sections synthesized based on different tracking strategies. 90

4.21 The comparison of boundaries in predicted sections synthesized by the tensor-based tracking method with or without noise adjusted. 90

5.1 The projection of neighboring samples on the transformation vector. 97

5.2 Comparison between TLPP and TOLDP-MMC. 98

5.3 A binary tensor sample contains gait silhouette. 105

5.4 The block diagram of the tracking method based on the OLDP-MMC. . . . 107

5.5 The texture tensor built from the boundaries of reference sections 107

5.6 Neighboring cubes defined around p in three directions 108

5.7 The reference image (inline 399) and its corresponding GoT map 109

5.8 The post-processing steps to synthesize the tracked boundary 111

5.9 The comparison between the green ground truth and the red tracked boundaries of inline 389 synthesized by (a) the tracking method based on the TOLDP-MMC and (b) the tracking method in Chapter 4 112

5.10 The details of the green ground truth and the red tracked boundaries in Figure 5.9 112

5.11 The SalSIM indices of tracked boundaries ranging from 389 to 395 and 403 to 409 113

SUMMARY

The identification of specific structures in three-dimensional (3D) data is an important task in a wide variety of fields such as cell detection in microscopic images[1], tumor segmentation in magnetic resonance images [2], and geological structure interpretation in migrated seismic data [3]. Understanding geological structures beneath the surfaces of the Earth, the Moon, or even Mars helps reveal underground hazards, protect groundwater aquifers, design man-made structures such as dams and tunnels, and explore mineral deposits. Hydrocarbon depositing beneath the Earth's surface is the world's most important source of energy. Products extracted or refined from hydrocarbon maintain the quality of our lives and the normal operations of modern society by supplying electric power to pubic and industry, heating homes, and providing fuel for vehicles and airplanes. Hydrocarbon exploration starts from seismic surveys, which fall into two categories, onshore and offshore. To acquire more geological information from data collected in seismic surveys, a series of processing steps are conducted to remove noise and artifacts and create the detailed images of various rock types and local geology in the subsurface. In processed seismic data, interpreters identify important geological structures such as faults, fractures, and salt domes, which are the implications of hydrocarbon reservoirs. The interpretation of these structures infers the geology in the subsurface and helps estimate the positions and sizes of hydrocarbon reservoirs. However, with the dramatic increase in data size, manual interpretation is becoming more time consuming and more labor intensive. Computational interpretation tools with high efficiency have been an urgent need in recent years.

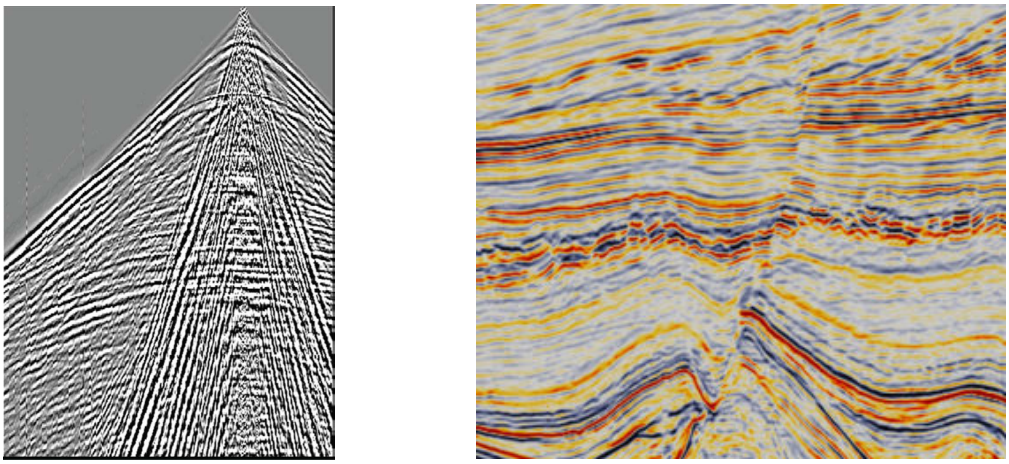
In this dissertation, we design a framework that interactively delineates important geological structures such as faults and salt domes in seismic volumes. Within the seismic volume, we divide seismic sections into two classes, reference and predicted ones. In reference sections, we implement interpreter-assisted workflows that delineate faults and salt domes using their geometric representations and texture features. Then, we introduce au-

tomated tracking workflows that label target structures in predicted sections by tracking corresponding delineated structures in reference sections through the seismic volume. To ensure accurate tracking, we extract features along geological structures using tensor-based subspace learning methods. Experimental results on real seismic data sets show that our interactive interpretation framework outperforms state-of-the-art methods in terms of accuracy and efficiency.

CHAPTER 1
INTRODUCTION AND BACKGROUND

Hydrocarbon exploration is an expensive and risky process. Drilling a well nowadays costs several millions of dollars [4], but no one can guarantee its hydrocarbon production. In addition, hydraulic fracturing involved in well drilling may lead to various environmental problems such as air and water pollution, soil and oil spill contamination, and even earthquakes. To reduce these risks, before drilling the potential profitability of a well and its damage to environment need to be evaluated as accurate as possible . Therefore, collecting information from Earth's subsurface is of great importance for hydrocarbon exploration. Subsurface imaging by seismic surveys produces the detailed images of various rock types and local geology. In a seismic survey, sound waves from seismic sources such as vibrator trucks (onshore) or air-guns (offshore) travel through the earth and bounce off subsurface layers. After reflected back to the surface, these waves are captured by receivers that are recording sensors such as geophones (onshore) or hydrophones (offshore).

Recorded data in the seismic survey, referred to as raw seismic data, can be manipulated into a new format that infers subsurface rocks and structures after a series of processing steps [5] such as deconvolution [6, 7], velocity analysis [8], stacking [9], and migration [10, 11]. Since migration is the last step in the seismic data processing pipeline, the processed data in the new format are commonly referred to as migrated data. Figure 1.1 shows the examples of raw seismic traces and one migrated seismic section. We notice that in Figure 1.1(a) raw seismic traces construct a slope since traces collected by receivers located near the seismic source have shorter travel time. Meanwhile, these collected traces describe only a tiny region in the subsurface, which cannot provide the intuitive illustrations of geological structures. In contrast, Figure 1.1(b) shows one migrated section, where we can distinguish the difference among various geological structures even without any geo-



(a) Raw seismic traces (b) Migrated seismic section

Figure 1.1: Examples of raw seismic traces and a migrated seismic section.

logical knowledge. In migrated seismic data, a trained interpreter can delineate important geological structures that are closely related to hydrocarbon traps such as faults and salt domes and further estimate the locations and sizes of hydrocarbon reservoirs. A successful manual seismic interpretation requires not only geological knowledge, but also substantial experience. With the dramatically growing sizes of seismic data collected in today's hydrocarbon exploration, manual interpretation has become more time consuming and more labor intensive. For example, a single seismic survey can generate up to one terabyte of data daily, and seismic exploration data sets can easily reach several petabytes [12]. If manual interpretation is the only available tool for an interpretation team, interpreters may have to spend months and possibly years on labeling and locating all important structures in such a dataset.

To improve efficiency and effectiveness, interpreters have begun utilizing computer programs to facilitate the interpretation process over the last decade [3]. However, imitating an experienced interpreter possessing sufficient geological knowledge is almost impossible for a computer because of complicated details in the subsurface. Although the implementation of fully automatic interpretation is difficult, computer programs can extract the quantitative features of various geological structures and *assist* manual seismic interpretation. Some

software platforms [13, 14] have successfully demonstrated their capability of reducing both time and labor cost involved in interpretation tasks. Moreover, to increase interpretation accuracy and robustness, computational interpretation methods commonly need to involve interactivity with interpreters. In most cases, a user-friendly interactive process may help avoid spending extensive time on tuning parameters. Therefore, computational interpretation with limited human intervention has been gaining more space in practice compared to the classical manual interpretation. Recently, modern seismic interpretation tools and research directions have moved towards image processing and machine learning algorithms.

For years, image processing theories and algorithms have been employed to assist structural interpretation and made essential contributions to the field. Structural interpretation commonly involves two main steps, attribute extraction and structure identification. Attributes that capture the key components of seismic data are implicitly or explicitly derived from signal and image processing techniques. For examples, instantaneous attributes [15] are derived using the Hilbert transform, and spectral attributes [16, 17] are the results of multi-resolution analysis such as continuous wavelet [18] and curvelet transforms [19]. To identify geological structures, edge, texture, and shape information, which are important for describing objects in natural images, are also applicable in seismic images. Edge detectors [20, 21], texture descriptors [22, 23], and the Hough transform [24, 25, 26] initially designed for nature images have showed their strong capability of identifying geological structures in seismic images. In addition, interpretation as a complex and subjective task relies on the human visual system (HVS). Recently, algorithms involving the HVS model such as saliency detection [27] and color space analysis [28] have been proposed to mimic the interpretation process by extracting the most perceptually representative features of seismic data. As we introduced above, the richness and fast progress in image processing and computer vision have taken the automation of structural interpretation to a higher level. Meanwhile, we believe that seismic interpretation as a challenging problem will continue

to impact the advancement of image processing in the future.

In recent years, the development of machine learning has shed a new light on its role in seismic interpretation by helping geologists understand the relationships between the large amounts of geological data or information. Machine learning models trained on input data facilitate seismic interpretation by producing repeatable and reliable results and alleviate two significant issues that interpreters may encounter, interpreting large volumes of data and understanding the relationship of various types of data simultaneously. Machine-learning-based interpretation methods have two main schemes. One extracts multiple seismic attributes based on the domain knowledge and experience of interpreters and then trains them on machine learning models, which could be either supervised or unsupervised, such as k-means clustering [29], the Gaussian mixture model [30], the multi-layer perceptron (MLP) [29, 31], and the support vector machine (SVM) [29]. The other conducts attribute extraction and structure classification on machine learning models with the input of local patches [32, 33], which has proven to be effective on analyzing visual imagery. Instead of extracting predefined multiple attributes, patch-based learning models are capable of building the mapping relationships between post-stack amplitude and structure spaces and automatically generating a group of features by taking local seismic reflection patterns into account. Therefore, the random or coherent seismic noise and processing artifacts involved in local patterns can be effectively identified and excluded. Without intervention by interpreters, patch-based learning models rely only on the appearance of geological structures in the dataset and mimic the behaviors of interpreters to a certain extent.

Computer-assisted interpretation on seismic volumes commonly has two strategies. One is to apply interpretation algorithms on two dimensional (2D) seismic sections. However, to achieve high interpretation accuracy, interpreters inevitably need to tune parameters on each section, which increases time and label costs on interpretation and reduces interpretation efficiency. The other is to extend interpretation algorithms designed for seismic sections to the 3D space and then apply them on seismic volumes. Although 3D interpre-

tation methods save time cost on tuning parameters, the time and space complexities of these algorithms are commonly higher than or at least equal to $\mathcal{O}(n^3)$, where n represents the edge length of a seismic volume regarded as a cube. Methods with such high computational complexities are time consuming. In addition, 3D interpretation methods aim to achieve accurate interpretation from a global viewpoint, which may ignore the details of local regions. In this dissertation, we make a trade-off between 2D and 3D seismic interpretation methods and design a novel framework for interactive interpretation on seismic volumes containing important geological structures such as faults and salt domes.

The rest of this dissertation is organized as follows. Chapter 2 conducts a literature survey of computational seismic interpretation and summarizes literature related to the computational interpretation of faults and salt domes. Chapter 3 presents a novel interactive interpretation framework that delineates faults in seismic volumes using geometric features extracted by the Hough transform. Chapter 4 extends the framework to the salt-dome interpretation and labels salt-dome boundaries using texture analysis and the tensor-based incremental learning method. Chapter 5 introduces a novel tensor-based subspace learning method and applies it to improve salt-dome tracking, which is a vital part in the interactive framework for salt-dome interpretation. Chapter 6 makes a conclusion for the entire dissertation.

CHAPTER 2

LITERATURE SURVEY

2.1 Computational Seismic Interpretation

The goal of computational seismic interpretation is to mimic the visual perception of experienced interpreters. Techniques that characterize the visual cues of migrated seismic data can be classified into three main directions, data representation, texture analysis, and the utilization of geological constraints.

2.1.1 Data Representation

A good representation of migrated seismic data is effective in revealing important geologic features that are not apparent for interpreters. In the 1970s, geophysicists defined subsurface regions with high reflection strength as "bright spots" [34], which indicate the possible locations of hydrocarbon reservoirs. Reflection strength that is independent of polarity and phase can not be simply represented by seismic traces. In [15], Taner *et al.* introduce an efficient measure of reflection strength, the trace envelop, which is the instantaneous amplitude of the complex seismic trace. The complex seismic trace has real and imaginary parts corresponding to seismic trace $x(t)$ and its Hilbert transform $y(t)$, respectively, in which t represents the travel time in the subsurface. Therefore, the instantaneous amplitude is calculated as $a(t) = \sqrt{x(t)^2 + y(t)^2}$. In addition, the instantaneous phase, $\phi(t) = \arctan[y(t)/x(t)]$, is the angle required to rotate $x(t)$ to $a(t)$. The instantaneous amplitude and phase together define more instantaneous attributes [35, 36] and form the foundation of several seismic attributes such as dip [37, 38] and coherence [39].

In addition to the complex trace representation, the time-frequency representation, which analyzes the seismic trace in both time and frequency domains simultaneously, has also

been widely used in computational seismic interpretation. Some geologic structures that are not apparent in the time domain can be identified in the frequency domain. For example, anomalies that are attributed to abnormally high attenuation in hydrocarbon reservoirs can be recognized in the low-frequency component [40]. Typical methods used for the time-frequency representation are the short-time Fourier transform (STFT) and the wavelet transform (WT). However, the main shortcoming in these methods is that they do not provide time resolutions in the frequency domain as high as interpreters expect. In recent years, more complicated methods have been proposed for time-frequency analysis such as the synchrosqueezing wavelet transform (SSWT) [41, 42] and multiple signal classification with empirical wavelet transform (MUSIC-EWT) [43].

2.1.2 Texture Analysis

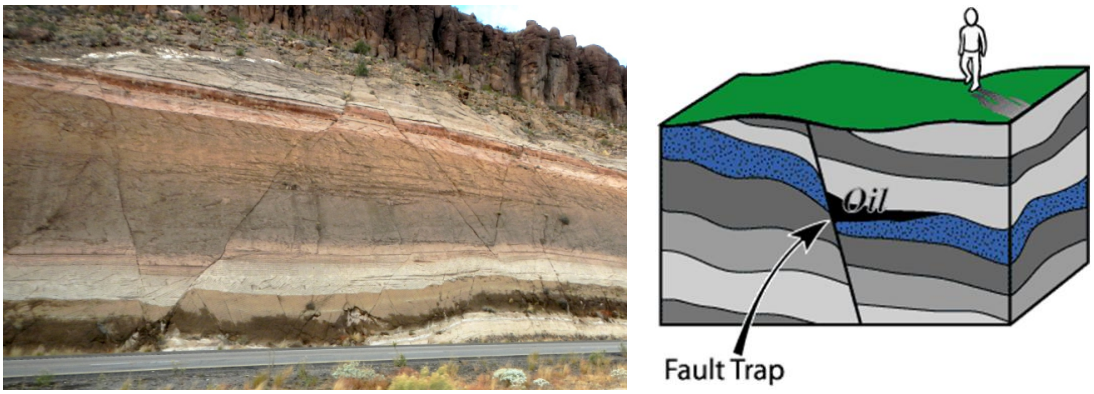
If we analyze a seismic section as an image rather than traces, we notice that different geological structures have different textures. For example, salt bodies have homogeneous textures. In contrast, the textures of horizons in two-dimensional (2D) seismic sections look like stripes with strong directionality. Textures, as a representative feature of geological structures, can be characterized by image analysis techniques. The gray level co-occurrence matrix (GLCM) [22] reflects the distribution of co-occurring grayscale values at a given offset. Statistics derived from the GLCM such as contrast, energy, and homogeneity can effectively describe textures from different perspectives. For example, the GLCM contrast describes the intensity difference between a pixel and its neighbors. Therefore, the contrast value of the pixel inside the salt body is close to 0. In addition, the GLCM homogeneity shows the homogeneous property of structures along the offset direction. Horizons have high homogeneity values if its dip is close to the offset direction.

The main disadvantage of GLCM-based attributes is their high computational complexity. In addition to the GLCM, semblance is also an important attribute for structures with strong discontinuity such as faults and fractures [44]. Semblance describes discontinuity

using the dip and amplitude information of the local neighborhood in seismic volumes. Recently, Shafiq et al. [45] have proposed an efficient texture attribute, the 3D gradient of texture (GoT), which quantitatively evaluates the texture dissimilarity of local neighboring volumes in a multi-scale manner. To imitate manual interpretation, the texture dissimilarity is analyzed based on human visual perception [46]. Similar to the semblance, the 3D GoT is sensitive to discontinuous structures and has shown its superior performance on salt-dome interpretation [45, 46, 47].

2.1.3 Geological Constraint

Geological structures formed under complicated conditions in the subsurface commonly have irregular shapes, which are hard to predict. However, geological constraints involved in the formation of subsurface structures may facilitate interpretation. For example, horizons formed by the deposition of rocks over different periods have strong continuity along dips. Considering such a constraint, interpreters identify horizons using the gradient structure tensor (GST) [48], which contains gradient information in all directions. The gradient normal to the horizon plane should be greater than those with directions parallel to the horizon plan. Therefore, the GST of horizons has one eigenvalue that are much greater than others, and its corresponding eigenvector determines the horizon dip [49]. In addition to horizons, faults formed under some geological constrains have the line-like or curved shapes in 2D seismic sections. Capturing the geometric features of faults is a main tendency in fault interpretation. More computational interpretation methods on faults and salt domes will be introduced in the following section.



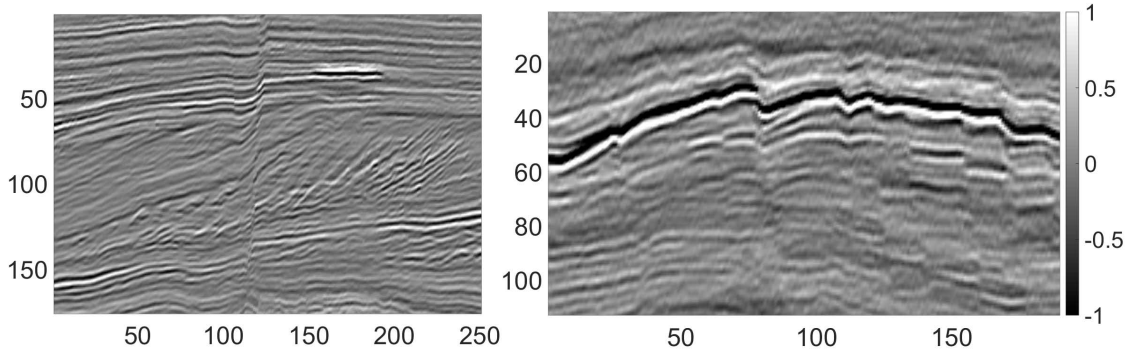
(a) (b)

Figure 2.1: (a) A normal fault near Kingman, Arizona [50], and (b) the illustration of a fault trap [51].

2.2 Literature Survey of Computational Interpretation on Fault and Salt-dome Structures

2.2.1 Fault Interpretation

A fault is defined as a lineament or planar surface across which apparent relative displacement occurs in the rocks layers. The movement of impermeable rocks and sediments along the fault surface creates membranes that hinder the migration of hydrocarbons from source rocks and form structural hydrocarbon traps. Therefore, faults are important geological implications for hydrocarbon exploration, and the understanding and analysis of the complicated relationships between fault networks and fractures is crucial for potential field development. Figure 2.1a shows a normal fault near Kingman, Arizona, and Figure 2.1b illustrates the structure of a fault trap. The formation process determines that faults have two distinct features. One is the geological feature, which is the discontinuity along horizons. The other is the geometric feature, i.e., line-like or curved shapes in 2D seismic sections, which appear as curved surfaces in 3D seismic volumes. Fault detection methods are commonly developed based on these two features. Figures 2.2a and 2.2b contain single and multiple faults, respectively. The discontinuity of faults can be characterized by several seismic attributes such as coherence [39], variance [52], curvature [53], similar-



(a) Single fault (b) Multiple faults

Figure 2.2: Examples of seismic sections containing single and multiple faults.

ity [54], entropy [55], and flexure [56]. Among them, coherence is the most popular one for highlighting faults. Similarly, Marfurt et al. [57] calculated semblance by comparing the dissimilarity of local regions on the two sides of a fault. Later, Gersztenkorn and Marfurt [58] proposed the eigenstructure-based coherence, referred to as C3 coherence, which analyzes the eigenstructure of covariance matrices of windowed seismic traces. Instead of calculating the eigenvalues of the covariance matrix, Yang et al. [59] proposed a computationally efficient coherence algorithm based on a normalized information divergence criterion. In addition, Li and Lu [60] combined spectral decomposition and complex coherence computation to map discontinuities at different scales. To avoid false low-coherence values in steeply dipping structures, Sui et al. [61] proposed a coherence algorithm that analyses the eigenstructure of the spectral amplitudes of seismic traces. More recently, Alaudah and AlRegib [62] have proposed a generalized-tensor-based coherence (GTC) attribute that derives covariance matrices from the unfolding matrices of a seismic analysis tensor along different modes corresponding to time, inline, and crossline directions, respectively. In contrast to C3 coherence, GTC enhances the details of discontinuity in seismic data. On the basis of GTC, Alaudah and AlRegib [63] implemented directional selectivity by involving a directional Gaussian preprocessing kernel and applying a 3D rotational matrix to the corresponding covariance matrix.

Although likely fault regions can be highlighted in attribute maps, under the influence of

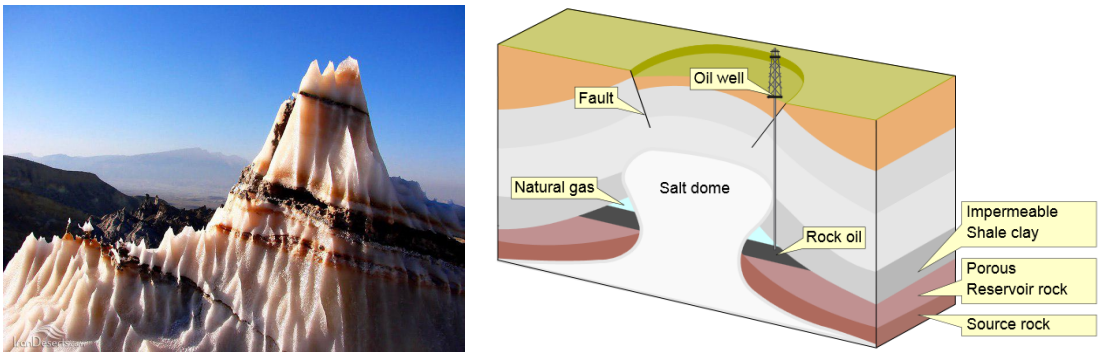
noise, the labeling of faults may not achieve reasonable accuracy. Therefore, enhancement operations such as non-linear mapping [64] and structure-oriented filtering [65, 66] have applied to increase the contrast between faults and surrounding structures. In addition, to improve accuracy and reliability, since 2000, researchers have proposed various interactive fault detection methods based on seismic attributes to delineate fault surfaces in seismic volumes or faults in 2D sections. By assuming that a fault is continuous and barely curved, Peterson et al. [67] proposed to utilize ant tracking or ant colony optimization algorithm to delineate fault surfaces. Similarly, Silva et al. [68] applied the ant-tracking algorithm on two seismic attributes, variance and chaos [69], to detects fault surfaces. In [70], Al-BinHassan et al. applied a smoothing operator on coherence cubes in order to remove noise and enhance faults. In the work of Cohen et al. [55], the proposed directional filters enhance the contrast of normalized differential entropy, and the skeletonization process [71] extracts one-pixel-width fault surfaces from likely fault regions. In [25], Jacquemin and Mallet proposed the cascaded Hough transform to roughly detect fault surfaces in seismic volumes. Different from the global detection of fault surfaces discussed above, Gibson et al. [72] designed a multi-stage approach that first highlights fault points in modified semblance cubes, generates local planar patches from grouped fault points, and finally merges small patches into large fault surfaces. In the work of Wang and AlRegib [73], the 3D Hough transform is utilized to detect small patches from the clouds of likely fault points, which are then merged to delineate the entire fault surface. Recently, Wu and Hale [74] proposed to use a simple linked data structure that includes the fault likelihood, dip, and strike to construct complete fault surfaces.

Because of the employment of 3D information, fault surfaces detected in seismic volumes reveal more accurate global structures of faults than those detected from 2D sections. However, fault interpretation methods applied on the entire seismic volume commonly have high time and space complexity, which requires powerful computational resources. To alleviate the limitation on computational power and simplify the detection process,

researchers have applied image processing and computer vision techniques on detecting faults from either seismic or time sections. Hale and Emanuel [75] proposed detecting faults on meshed time sections by involving typical image segmentation methods such as normalized cuts [76] and stochastic clusters [77]. In another work of Hale [66], he applied directional Gaussian filters to enhance semblance maps, selected fault points with the largest semblance values, and connected these points to label faults. Although this method is robust and highly automatic, false features involved in detected faults reduce labeling accuracy. Recently, Zhang et al. [78] have borrowed a biometric algorithm to detect faults in time sections under the assumption of a high resemblance between faults and the capillary veins of human fingers. In [28], using the semblance maps of every three neighboring time sections, Wang et al. synthesized RGB color images. By investigating the influence of the human vision system on seismic interpretation, the proposed method converts RGB color images to other color spaces, extracts likely faults from corresponding luminance components, and combines likely faults under geological and connectivity constraints. Because of the line shapes of faults in 2D seismic sections, in [26] AlBinHassan and Marfurt used the Hough transform to detect all lines in coherence maps computed from seismic amplitude maps. However, without noise rejection, this method labels only the rough shapes of faults. Therefore, to overcome this drawback, Wang and AlRegib [64] proposed a fault detection method that extracts the line features of faults using the Hough transform and removes noisy features with geological constraints.

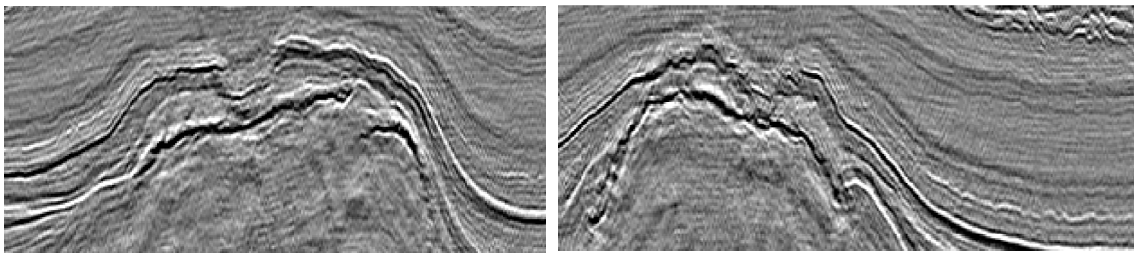
All fault detection methods introduced above were proposed based on typical image processing techniques, which commonly involve some necessary parameters. The selection of parameters determines interpretation performance. Because of the existence of parameters, it is difficult for these algorithms to simultaneously achieve both high recall and high precision when extracting faults. The common result is either an aggressive case with higher recall (most/all true faults extracted) but lower precision (many artifacts introduced), or a conservative one with higher precision (few artifacts introduced) but lower recall (few

true faults extracted) [79]. To alleviate this trade-off and increase robustness and generality, machine learning techniques have been involved in fault detection methods [80, 81] and become more popular in recent years. Di et al. [82] presented an innovative workflow based on the support vector machine (SVM) [83] to detect faults. First, three groups of seismic attributes including edge-based, geometric, and texture ones are extracted from the amplitude volume to highlight faults moderately. Second, the samples of faults and non-faulting zones are manually picked from the amplitude volume. Third, training the SVM model on the attributes of selected samples leads to an optimal binary classifier for volumetric processing. Finally, applying the trained SVM model on the entire seismic volume generates a binary volume, in which the presence of a fault is labelled as one. The accuracy of the multi-attribute-based classification workflow depends highly on the selection of seismic attributes, which should be capable of distinguishing faults and other geologic structures [84]. Because of the complexities of subsurface geology and the presence of seismic noise, most seismic attributes fail to have robust performance on highlighting faults. Without manually choosing seismic attributes, the convolutional neural network (CNN) [85] provides a way to learn certain seismic attributes characterizing faults from the appearance of faults in seismic data. Di et al. [86] proposed an attribute-free fault detection method using the CNN. Instead of using attributes as the input, the CNN-based method defines the input as local seismic reflection patches, which have been labelled as fault or non-faulting zones, respectively, according to the location of the central seed. Training the pre-defined CNN on local patches builds the mapping relationship between seismic signals and fault structures. The accurate detection of faults in [86] not only verifies the capability of the CNN in assisting fault interpretation, but also indicates greater potentials for the applications of more advanced networks (e.g., fully convolutional networks (FCN) [87] on structure interpretation. In addition, except for the labelling of training data, this whole scheme does not require the intervention of interpreters and is applicable to vast data sets without repeated efforts in attribute selection



(a) (b)

Figure 2.3: (a) Jashak salt dome, known as Dashti salt dome, is located in Dashti region of Bushehr province in Southern Iran [88], and (b) the illustration of a salt trap [89].



(a) (b)

Figure 2.4: Examples of salt domes in seismic sections.

2.2.2 Salt-dome Interpretation

A salt dome is defined as a dome-shaped structure formed by the evaporation of a large mass of salt in sedimentary rocks. Salt domes are impermeable structures that prevent the migration of hydrocarbons and provide entrapment for oil and gas reservoirs. Figure 2.3a shows Jashak salt dome located in Southern Iran, and Figure 2.3b illustrates the structure of a salt-dome trap. In migrated seismic data, salt domes containing chaotic reflections have different texture from surrounding structures formed by other types of rocks. Figures 2.4a and 2.4b illustrate two examples of salt domes in seismic sections. Therefore, salt-dome interpretation can be treated as a structure segmentation problem in the field of image processing and computer vision. However, the boundaries of salt domes are not easily to be identified in most geological scenarios because of the underlying physics as well

as noisy and low-resolution data. To characterize the boundaries of salt domes, in addition to typical seismic attributes introduced in the previous section, researchers have proposed various novel seismic attributes in recent years. Hegazy and AlRegib [90] derived a directional attribute from the moment of the inertia tensor for gradient components to describe the textures of salt bodies. Shafiq et al. [27] proposed a seismic attribute, referred to as SalSi, which highlights the salient areas of a seismic image by comparing local spectral features based on the 3D fast Fourier transform (FFT). Chopra and Marfurt [91] proposed a seismic disorder attribute to assess randomness and the SNR of data to delineate seismic structures such as faults and salt domes. In addition, Wu [92] proposed methods to compute salt likelihoods highlighting salt boundaries, extract oriented salt samples on ridges of salt likelihoods, and construct salt boundaries with selected salt samples by solving a screened Poison surface reconstruction problem.

By defining salt-dome detection as a segmentation task, some automated workflows have been proposed to involve image processing and computer vision techniques such as graph cuts, edge detection, and texture analysis. Lomask et al. [93] defined seismic sections as weighted undirected graphs, in which nodes are pixels while edges connect all pairs of nodes. The weight of an edge is determined by the spatial distance of two corresponding nodes and the similarity of their attribute intensity. Based on the normalized cut image segmentation (NCIS) [76], the delineation of salt-dome boundaries can be globally optimized. As an extension, Lomask et al. [94] added the local dip difference to the weight function and involved bound constraints to remove boundary artifacts. Similarly, Halpert et al. [95] applied the NCIS on multiple seismic attributes such as instantaneous amplitude, dip, and instantaneous frequency and combined corresponding segmentation results using adaptive weights. Although these NCIS-based methods can be implemented in parallel, the high computational cost will limit their future application on high-resolution or 3D seismic data. To improve the efficiency of global segmentation, Halpert et al. [96] modified the pairwise region comparison algorithm (Felzenszwalb and Huttenlocher [97]) using the

refined graph structure and weight function. In contrast to NCIS-based methods with the time complexity of $\mathcal{O}(n^2)$, the method based on pairwise region comparison involves the minimum spanning tree and reduces time complexity to $\mathcal{O}(n \log n)$, where n is the number of pixels.

Different from graph cuts, edge detection techniques are simple and have high computational efficiency. In recent years, methods based on edge detectors have been proposed to delineate the boundaries of salt domes. Jing et al. [98] applied the 2D Sobel filter, a simple formulation of derivatives with weighted masks, on post-stack time sections to highlight salt-dome boundaries. Without amplitude normalization, the scheme works well only when seismic data exhibits small amplitude variations in two directions. To enhance salt-dome boundaries in different directions, Aqrawi et al. [99] proposed a salt-body detection algorithm based on the 3D Sobel detector, which involves both amplitude normalization and dimension weighting. More recently, Amin and Deriche [21] proposed an approach that highlights small variations in seismic data by detecting edges not only along the x, y, and z directions but also slanted at $45°$and $-45°$ using 3D Sobel filters.

As we introduced in the previous section, automated workflows based on image processing techniques focus on describing the appearance of geological structures and are not robust enough to complex geological scenarios. The inevitable parameter tuning requires extra time and limits automation efficiency. Similar to fault detection, applying machine-learning techniques on salt-dome detection has become a new trend in recent years. In the work of Berthelot et al. [30], a feature vector is constructed for each pixel from various attributes, including GLCM-based attributes [22], frequency-based attributes [100], and typical seismic attributes such as the coherence and dip [44, 101]. Based on extracted features, the proposed method divides pixels into four classes (salt body, sub-horizontal, up-dipping, and down-dipping reflectors) and applies a supervised Bayesian classification model to extract salt domes. Similarly, Amin et al. [102] computed the GLCM- and GoT-based attributes [22, 103] of patches centered at pixels and applied a dictionary-based classifier to

distinguish salt-dome boundaries and surrounding non-boundary structures.

A workflow proposed by Guillen et al. [104] automatically detects salt domes from the SEG-SEAM dataset using the extremely random trees ensemble [105] to classify attributes. More recently, based on the k-means clustering [106] of multiple seismic attributes, Di et al. [107] proposed an interpreter-assisted approach to highlight salt-dome boundaries in the F3 block. To investigate the performance of various classification methods on salt-dome detection, Di and AlRegib [108] trained six classification models including the logistic regression, the decision tree, the random forest, the SVM, the artificial neural network, and k-means clustering on multiple attributes including RMS amplitude, GLCM-based attributes [22], the GoT [103, 109], seismic saliency [27], and Canny edge detector [110]. In [108], the good match between detected salt boundaries and original seismic images indicates that with well-selected attributes all six classification techniques are capable of providing reliable salt detection from 3D seismic data. To avoid the manual selection of seismic attributes, which requires the domain knowledge and experience of interpreters, Di et al. [111] proposed a CNN-based workflow to implement the salt-dome delineation in the SEG-SEAM data set, which shows the superiority on delineating salt domes in a challenging scenario.

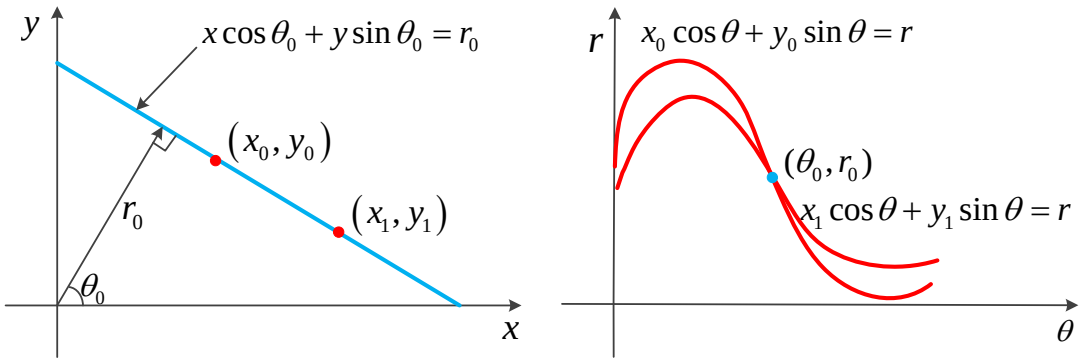
CHAPTER 3
THE ROLE OF GEOMETRIC REPRESENTATION IN INTERACTIVE FAULT INTERPRETATION

The movement of neighboring rock layers may lead to significant displacements along fractures in the subsurface and form the important geological structure, faults. As effective structural traps, faults may seal reservoir rocks and contribute to the formation of petroleum reservoirs. As we introduced in the previous section, the formation process determines that faults have two distinct features. One is the geological feature, which is the discontinuity along horizons. The other is the geometric feature, i.e., line-like or curved shapes in 2D seismic sections, which appear as curved surfaces in 3D seismic volumes. In [25, 26], the Hough transform has been shown to be a powerful tool for detecting the geometric features of fault detection. In this chapter, we develop a method that describes the geological features of faults using the discontinuity attribute, extracts the geometric features of faults using the Hough transform, and implement the interactive labelling of faults using tracking vectors. This chapter consists of three sections. We firstly introduce 2D and 3D Hough transforms as well as the concept of tracking vectors. Then we introduce the interactive fault interpretation method in details. Finally we make a summary of this chapter.

3.1 Hough Transform and Motion Vectors

3.1.1 2D Hough Transform

In the field of image processing, the 2D Hough transform [24], a mapping procedure between image and the parameter spaces, has been widely used to detect the lines, circles, and other parametric curves. Figure 3.1 illustrates a simple example of the 2D Hough transform. In Figure 3.1a, we analytically describe the line in the image space with its



(a) Image space (b) Parameter space

Figure 3.1: The 2D Hough transform represents a mapping procedure between the image space and the parameter space.

parametric equation as follows:

$$x \cdot \cos\theta_0 + y \cdot \sin\theta_0 = r_0, \tag{3.1}$$

where r_0 represents the algebraic distance [112] between the line and the origin and θ_0 defines the orientation of r_0 with respect to x-axis. If we restrict θ_0 to the interval of $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, parameter pair (θ_0, r_0), which identifies a line in the image space, represents a unique point in the parameter space as Figure 3.1b shows. Suppose that we have two arbitrary points (x_0, y_0) and (x_1, y_1) in the image space and try to find a line fitting them. To solve this problem, we transform two points to the parameter space, in which (x_0, y_0) and (x_1, y_1) as parameter pairs define two sinusoidal curves. The corresponding equations of sinusoidal curves are shown as follows:

$$\begin{cases} x_0 \cdot \cos\theta + y_0 \cdot \sin\theta = r \\ x_1 \cdot \cos\theta + y_1 \cdot \sin\theta = r \end{cases}. \tag{3.2}$$

In Figure 3.1b, these two curves intersect at point (θ_0, r_0), which describes the blue line in the image space passing through points (x_0, y_0) and (x_1, y_1) as Figure 3.1a shows. Therefore, when we have a group of points in the image space, the problem of finding a set of

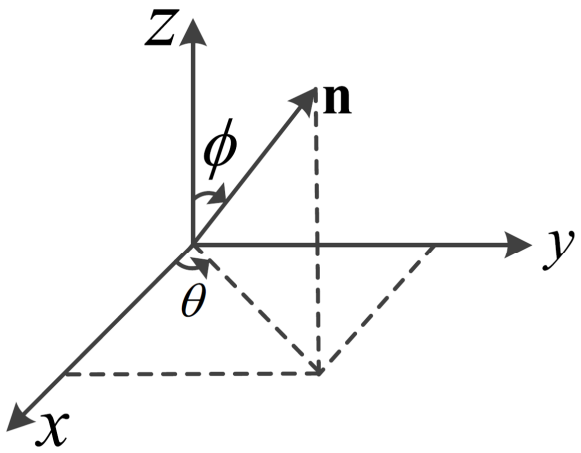


Figure 3.2: The normal vector in 3D spherical coordinate system.

lines fitting them can be converted to the problem of identifying points in the parameter space, where the largest number of sinusoidal curves intersect. We summarize the properties of the 2D Hough Transform for line detection as follows:

1. A point in the image space corresponds to a sinusoidal curve in the parameter space.
2. A point in the parameter space corresponds to a line in the image space.
3. Points on the same line in the image space corresponds to sinusoidal curves passing through a common point in the parameter space.
4. Points on the same curve in the parameter space corresponds to lines passing through a common point in the image space.

3.1.2 3D Hough Transform

As an extension of the 2D Hough transform, the 3D Hough transform is able to detect planes and 3D parametric curves in 3D point clouds [113]. In the 3D space a plane is defined by a normal vector and a known point on the plane. Figure 3.2 illustrates an example of unit vector $\mathbf{n}$ in the spherical coordinate system, where θ represents the azimuthal angle with a range from 0 to 2π and ϕ is the polar angle with a range from 0 to π. The expression

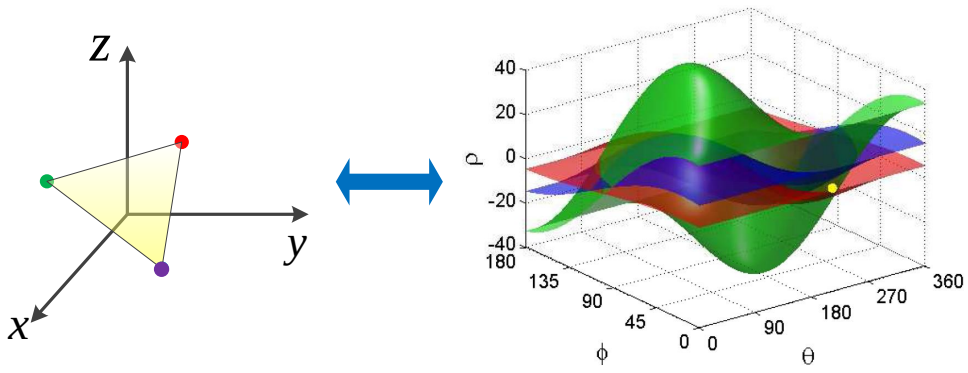


Figure 3.3: The mapping relationship between volume and parameter spaces in the 3D Hough transform.

of $\mathbf{n}$ in the form of θ and ϕ is shown as follows:

$$\mathbf{n} = (x, y, z) = (\sin\phi\cos\theta, \sin\phi\sin\theta, \cos\phi)\,. \tag{3.3}$$

With unit normal vector $\mathbf{n} = (\sin\phi_0\cos\theta_0, \sin\phi_0\sin\theta_0, \cos\phi_0)$ and known point $\mathbf{r_0} = (x_0, y_0, z_0)$ on the plane, we obtain the equation of the defined plane as follows:

$$\begin{cases} \sin\phi_0\cos\theta_0 \cdot x + \sin\phi_0\sin\theta_0 \cdot y + \cos\phi_0 \cdot z = \rho \\ \rho = \mathbf{n} \cdot \mathbf{r_0} = \sin\phi_0\cos\theta_0 x_0 + \sin\phi_0\sin\theta_0 y_0 + \cos\phi_0 z_0 \end{cases}, \tag{3.4}$$

where ρ is the algebraic distance [112] from the original point to the plane.

Similar to the 2D Hough transform, the 3D Hough transform builds a mapping relationship between volume and parameter spaces. According to the parametric equation in Eq. (3.4), one plane in the volume space is defined by θ_0, ϕ_0, and ρ_0, which represents one unique point in the parameter space. Suppose that we have three arbitrary non-collinear points in the volume space and try to find a plane fitting them. To solve this problem, in Figure 3.3 we transform all three points to the parameter space, in which (x_i, y_i, z_i), $i = 1, 2, 3$, as parameter pairs define three sinusoidal surfaces as the following equation shows:

$$\sin\phi\cos\theta \cdot x_i + \sin\phi\sin\theta \cdot y_i + \cos\phi \cdot z_i = \rho, \quad i = 1, 2, 3. \tag{3.5}$$

Surfaces in green, purple, and red pass through a common yellow point, which represents the plane in the volume space determined by three non-collinear points. Therefore, to find a set of planes fitting point clouds in the volume space, we need to search for points in the parameter space, where the largest number of sinusoidal surfaces intersect. However, searching complexity in the 3D space is much higher than that in the 2D space. Considering the automation and efficiency of fault interpretation, we utilize the 2D Hough transform to detect fault features.

3.1.3 Tracking Vectors

A video can be identified as a sequence of frames. In the same scene content, the successive frames of a video are almost the same except for small differences. Such redundancy of information existing between successive frames is temporal redundancy. Reducing temporal redundancy as a primary techniques in video coding standards is typically implemented by defining three types of frames: I-, P-, and B-frames. We list the features of various frames as follows:

1. I-frames as "key/reference frames" with low compression ratios have no reference to other frames.

2. P-frames as "predicted frames" are predicted from previous reference frames, which could be either I- or P-frames. P-frames have higher compression ratios than I-frames, since only differences with reference frames are stored. Therefore, P-frames cannot be reconstructed without reference frames.

3. B-frames as "bidirectional frames" are the bidirectional version of P-frames and interpolated from previous and forward reference frames. B-frames have the highest compression ratios and cannot be referenced by other P- or B-frames

Reference relationships between frames are commonly implemented by two processes, motion estimation and compensation. Motion estimation acquires motion vectors from

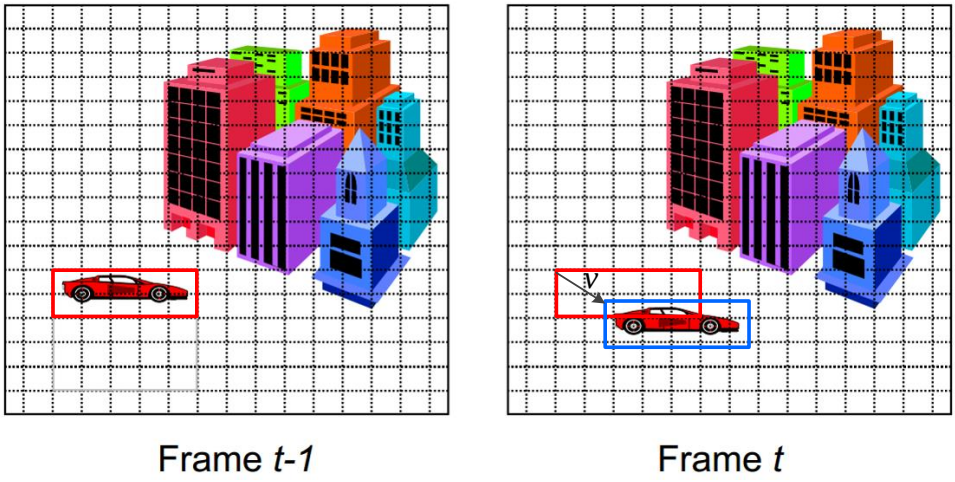


Figure 3.4: Motion vector v between two successive frames $t-1$ and t.

successive frames by analyzing the movement of objects. With motion vectors, motion compensation identifies matched regions in successive frames, which have strong similarity. Pixel differences between matched regions as error residuals have a majority of zero or close to zero values, which require less bits for coding and increases compression ratios. Figure 3.4 illustrates an example of a motion vector in two successive frames. Motion vector v describes the movement of the car from frame $t-1$ to frame t. Although faults cannot move in a seismic volume, strong correlations between neighboring seismic sections are similar to those between the successive frames of videos. Therefore, we employ the concept of motion vectors for fault tracking and define tracking vectors to describe the location changes of faults. More details about tracking vectors will be introduced in the following section.

3.2 Interactive Fault Interpretation Framework

In this section, we develop a method that interactively labels faults using the Hough transform and tracking vectors. The diagram of the interactive interpretation method is shown in Figure 3.5. In the preprocessing step, we calculate the coherence of seismic volumes, which enhances discontinuous structures such as faults and fractures. From likely fault regions

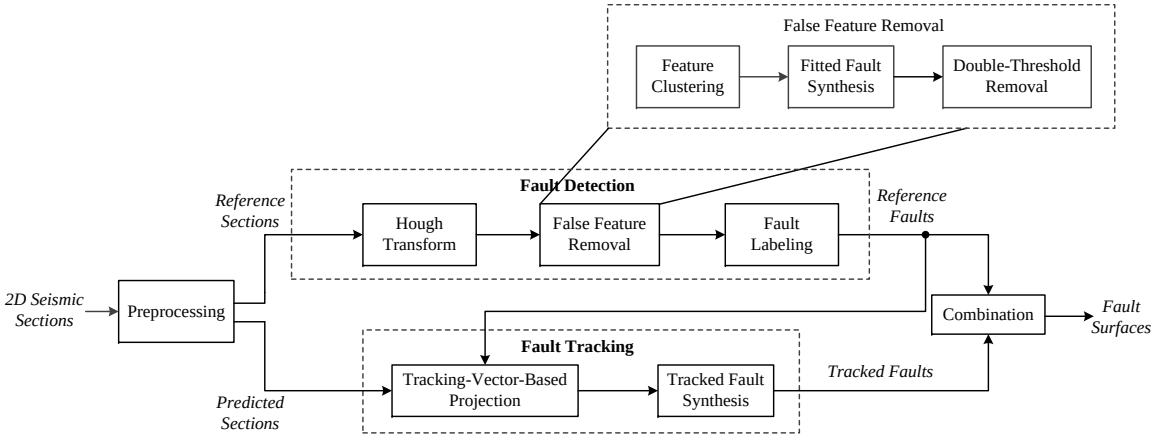


Figure 3.5: The interactive fault interpretation method has two main parts, fault detection in reference sections and fault tracking in predicted sections.

highlighted in coherence maps, we detect the feature of faults using the Hough transform, which is a powerful tool for describing geometric shapes as we discussed above. However, if interpreters depend only on such a detection method, they have to spend extra time on adjusting parameters when dealing with different seismic sections. To improve interpretation efficiency, we attempt to utilize strong correlations between neighboring seismic sections by borrowing the idea of I- and P-frames from video coding and processing techniques. We divide seismic sections into two different groups, reference and predicted sections. Since reference sections account only for a small proportion of the whole seismic volume, to ensure fault interpretation accuracy, we detect the faults of reference sections using the Hough transform. In contrast, we synthesize tracked faults in predicted sections on the basis of estimated tracking vectors and the projections of detected faults from neighboring reference sections. The involved fault information from reference sections simplifies the tracking process and ensures it can be implemented almost automatically. Finally, by combining detected and tracked faults, we yield fault surfaces in the 3D seismic volume.

3.2.1 Preprocessing

In the preprocessing step, in order to characterize faults more accurately and efficiently, we acquire a prominent feature of faults, lateral discontinuity. This attribute is derived from

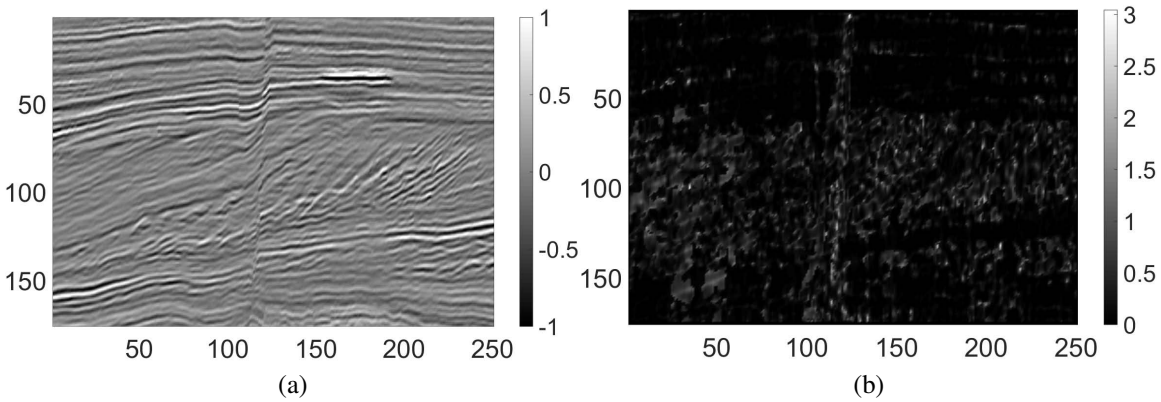
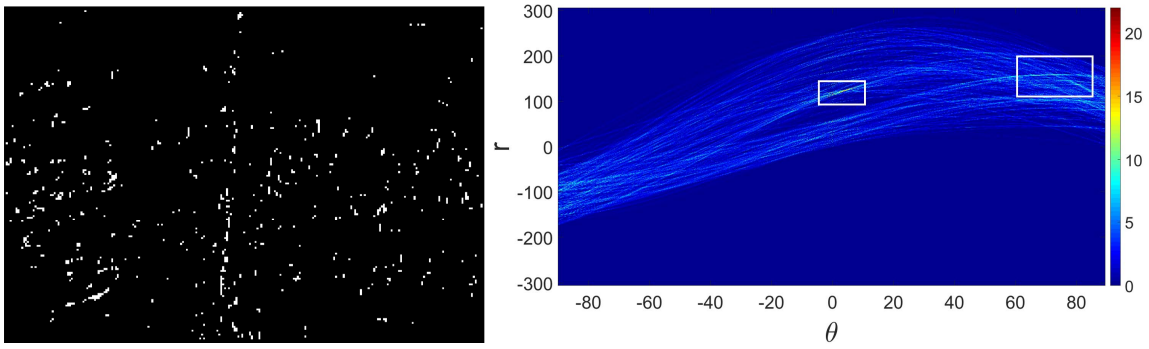


Figure 3.6: An example of a seismic section (Inline #268) and its corresponding discontinuity map.

the coherence attribute proposed by Manfurt et al. [44], which outperforms other attributes in the identification of faults by taking local dip information into account. In the pixel-wise calculation of the coherence-based attribute, every point in seismic sections represents the center of an analysis window that is oriented along the local horizon. By involving all neighboring pixels in a $(2r_d+1)\times(2r_d+1)$ analysis window, we obtain the corresponding discontinuity value $D(x,z)$, calculated as follows:

$$D(x,z)=\left|\ln\left[\frac{\sum_{j=-r_d}^{r_d}\left(\sum_{i=-r_d}^{r_d}S(x+i,z+j)\right)^2}{(2r_d+1)\sum_{i,j=-r_d}^{r_d}S(x+i,z+j)^2}\right]\right|, \tag{3.6}$$

where x and z correspond to the crossline and depth direction, respectively, and $S(x,z)$ represents the intensity of seismic signals. In addition, the function $|\ln(\cdot)|$ ensures the nonnegativity of discontinuity values and increases the contrast between faults and coherent features. Therefore, points with greater discontinuity values have the higher possibility of being located in faults. In contrast, points, which have discontinuity values close to zeros, correspond to coherent features. Figure 3.6 shows an example of a seismic section and its corresponding discontinuity map. In Figure 3.6a, the intensity values of seismic images have been normalized between -1 and 1. In Figure 3.6b, we notice that bright regions correspond to likely fault regions, and dark regions indicate coherent features.



(a) (b)

Figure 3.7: (a) Binary image B containing likely fault points; (b) sinusoid curves mapped from fault points

3.2.2 Fault Detection in Reference Sections

Line Feature Extraction Using the Hough Transform

On the basis of calculated discontinuity maps, we attempt to label faults in a reference section using the Hough transform. Since the shapes of faults are commonly lines and flat curves in seismic sections, we can extract the line features of faults from likely fault regions using the Hough transform. To identify likely fault regions, we apply threshold T_H on the discontinuity map and obtain binary image B as follows:

$$B(x,z) = \begin{cases} 1, & \text{if } D(x,z) \geq T_H \\ 0, & \text{otherwise} \end{cases}. \tag{3.7}$$

Figure 3.7a shows binary image B obtained based on the binarization of the discontinuity map in Figure 3.6b. Furthermore, we transform every likely fault point to the corresponding sinusoidal curve in parameter space θ-r and identify points where the largest number of curves intersect. In Figure 3.7b, color values represent the number of intersecting curves. Therefore, we select intersections from green areas labeled by white rectangles. Finally, by mapping these selected intersections back onto the image space, we extract line features from the discontinuity map. In the procedure of the Hough transform, interpreters need to

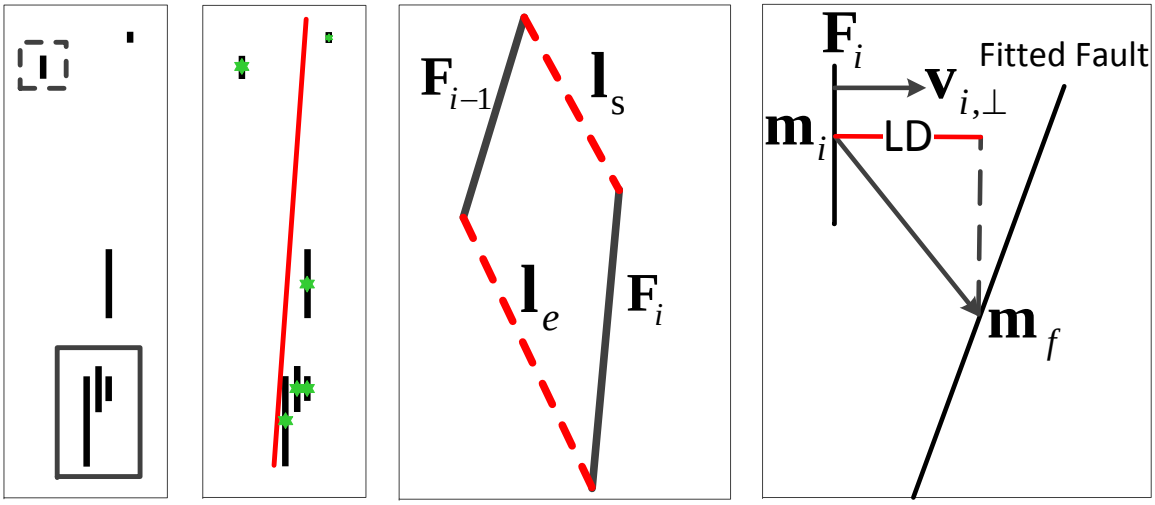


Figure 3.8: The process of false feature removal: (a) types of false features, (b) a fitted fault based on the midpoints of all lines, (c) the illustration of *absolute distance*, and (d) the illustration of *lateral distance*.

adjust two important parameters. One is the number of selected intersections, which determines the number of detected lines in the image space. The other parameter is the slope interval of detected lines, which is constrained by fault orientations. However, interpreters can avoid spending time on optimally adjusting parameters since incoherent anomalies in extracted line features will be removed in the following step.

False Feature Removal

Because of the limitation of the Hough transform, it is inevitable that detected results contain some false features that violate geological constraints. According to the locations of false features, we classify them into two types as shown in Figure 3.8a. The line inside the dashed box, which is isolated from the others, is an *outlier*, since the line features of faults should appear around fracture surfaces in natural scenarios. In addition, the lines labeled by the solid box, located in close proximity, represent a *neighboring group*, which describe the same fault region and should be merged into one. To filter out these false features, we introduce a double-threshold method with a diagram shown at the top of Figure 3.5. Before

we describe the method in detail, we need to first define detected lines using 2×2 matrices as follows:

$$\mathbf{F}_i = \begin{bmatrix} x_{s,i} & z_{s,i} \\ x_{e,i} & z_{e,i} \end{bmatrix}, i = 1, 2, \cdots, N_F, \tag{3.8}$$

where $(x_{s,i}, z_{s,i})$ and $(x_{e,i}, z_{e,i})$ represent the coordinates of the starting and ending points of $\mathbf{F}_i$, respectively. The indices, $i = 1, 2, \cdots, N_F$, increase with the depth positions of detected lines in seismic sections, and N_F represents the total number of line features. In addition, we denote the midpoint of line $\mathbf{F}_i$ as

$$\mathbf{m}_i = \left[\frac{x_{s,i} + x_{e,i}}{2}, \frac{z_{s,i} + z_{e,i}}{2} \right], i = 1, 2, \cdots, N_F. \tag{3.9}$$

In Figure 3.8b, stars indicate the midpoints of all lines.

Since one seismic section may contain several faults, first of all, we must classify the detected lines into different groups and ensure that lines in each group belong to the same fault. To solve this clustering problem, we apply the K-means clustering algorithm on the midpoints of all lines and obtain the optimal partition by tweaking initial means. Furthermore, to remove *outliers*, in each group, we need to define the distance between one feature and the others. Therefore, we conduct the linear regression analysis of all midpoints and generate a fitted line that roughly describes the position of a real fault. As Figure 3.8b depicts, the red line is fitted from all green midpoints.

After grouping features and synthesizing fitted faults, to determine whether one detected line is a false feature or not, we define two distances, *absolute distance* (AD) and *lateral distance* (LD). *Absolute distance*, representing the position relationship between two neighboring features is calculated as

$$AD = \frac{1}{\sqrt{2}} \left\| \mathbf{F}_{i-1} - \mathbf{F}_i \right\|_F, \tag{3.10}$$

where $\|\cdot\|_F$ represents the Frobenius norm. The geometric definition of AD is derived

from the quadratic mean of lines $\mathbf{l}_s$ and $\mathbf{l}_e$, which connect the starting and ending points of $\mathbf{F}_{i-1}$ and $\mathbf{F}_i$, respectively, as Figure 3.8c illustrates. In contrast, *lateral distance*, labeled in Figure 3.8d, describing the offset distance between one feature and the fitted fault, is geometrically defined as the projection of the vector connecting $\mathbf{m}_i$ and $\mathbf{m}_f$ onto the direction perpendicular to $\mathbf{F}_i$, as the following equation shows

$$LD = \left| (\mathbf{m}_i - \mathbf{m}_f) \cdot \mathbf{v}_{i,\perp} \right|, \tag{3.11}$$

where $\mathbf{m}_f$ is the midpoint of the fitted fault and $\mathbf{v}_{i,\perp}$ represents the unit vector perpendicular to $\mathbf{F}_i$. Using the pre-defined AD and LD, the double-threshold method removes false features, the details of which are provided in the following pseudo-code:

Algorithm 1 False feature removal

```
for i ← 1 to N_F do
    if LD ≥ T_L then
        F_i ← F_{i−1}
    else
        if AD ≤ T_A then
            F_i ← argmax_{x∈{F_{i−1}, F_i}} L(x)
        end if
    end if
end for
```

where function $L(\cdot)$ calculates the length of detected lines and T_L and T_A are two thresholds employed for removing false features based on obtained LDs and ADs. If the LD of $\mathbf{F}_{i-1}$ is greater than T_L, it indicates that $\mathbf{F}_{i-1}$ is an *outlier* isolated from the other lines and needs to be discarded. In addition, if the AD of $\mathbf{F}_{i-1}$ is less than T_A, it suggests that $\mathbf{F}_{i-1}$ and $\mathbf{F}_i$ belong to a *neighboring group* and need to be merged into a longer one. After examining all detected lines from top to bottom and removing false ones, we obtain the remaining lines shown in Figure 3.9a.

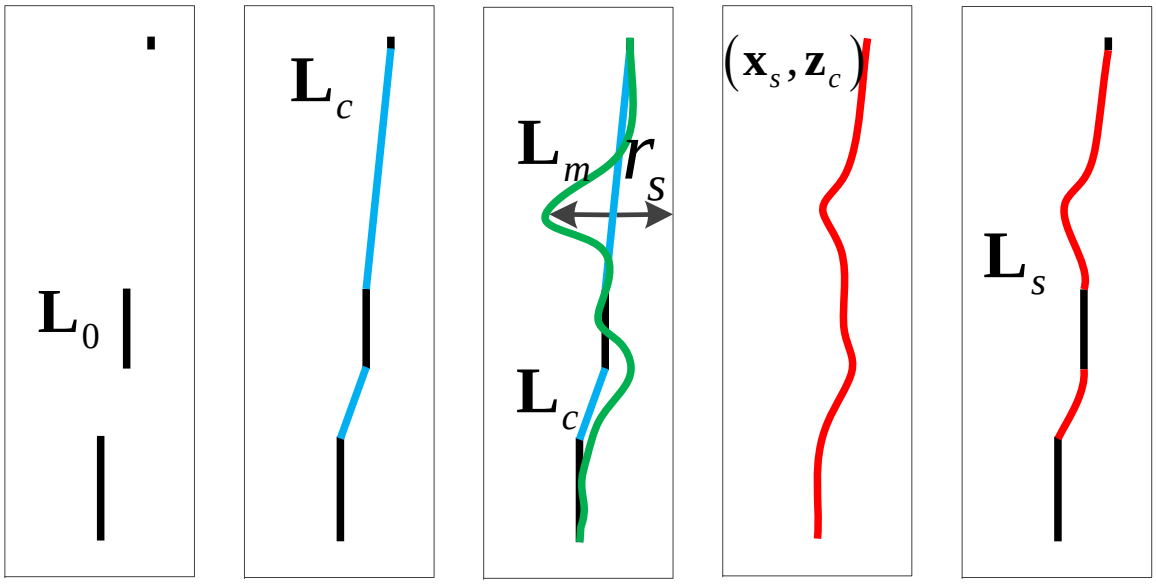


Figure 3.9: The process of fault labeling: (a) the remaining lines after false feature removal, (b) initial labeling, (c) the searching results based on the discontinuity map, (d) optimized labeling, and (e) the labeled fault.

Fault Labeling

On the basis of the remaining lines, we attempt to label faults as accurately as possible. Before introducing our labeling strategy, we first define matrix $\mathbf{L}_0 = (\mathbf{x}_0, \mathbf{z}_0)$, in which $\mathbf{x}_0$ and $\mathbf{z}_0$ contain the x and z coordinates of all points in the remaining lines, respectively. The most straightforward way to delineate faults is to use additional lines to connect these remaining lines in order. As illustrated in Figure 3.9b, the remaining lines connected by blue lines depict the initial shape of the fault. However, since we connect the remaining lines without involving any geological information, the initial labeling, denoted $\mathbf{L}_c = (\mathbf{x}_c, \mathbf{z}_c)$, is not accurate.

To obtain a more accurate delineation of faults, we need to employ discontinuity maps that characterize the geological features of faults. In point-wise labeling, we denote the coordinates of points in $\mathbf{L}_c$ as $(\mathbf{x}_c(j), \mathbf{z}_c(j))$, $j = 1, 2, \cdots$. Since faults in seismic sections are near-vertical, to avoid damaging the structures of the labeling results, for each $\mathbf{z}_c(j)$,

we search along the crossline (horizontal) direction at radius r_s and identify new fault point $(\mathbf{x}_m(j), \mathbf{z}_c(j))$ with the greatest local discontinuity value. Eq. (3.12) demonstrates this searching process as follows:

$$\mathbf{x}_m(j) = \underset{x \in [\mathbf{x}_c(j)-r_s, \mathbf{x}_c(j)+r_s]}{\arg\max} D\left(x, \mathbf{z}_c(j)\right). \tag{3.12}$$

The green curve in Figure 3.9c represents searching result $\mathbf{L}_m = (\mathbf{x}_m, \mathbf{z}_c)$. Therefore, we have two candidates for labeling faults. One is $\mathbf{L}_c$ from the initial labeling, and the other is $\mathbf{L}_m$ from Eq. (3.12). Since both candidates have the same depth coordinates $\mathbf{z}_c$, for each $\mathbf{z}_c(j)$, we need to determine the position of the fault point between $\mathbf{x}_c(j)$ and $\mathbf{x}_m(j)$. To measure the relative influence of $\mathbf{x}_c$ and $\mathbf{x}_m$, we have an objective function as follows:

$$\mathbf{x}_s = \underset{\mathbf{x}}{\arg\min}\ \lambda_c \|\mathbf{x} - \mathbf{x}_c\|_2^2 + \lambda_m \|\mathbf{x} - \mathbf{x}_m\|_2^2, \tag{3.13}$$

where λ_c and λ_m, which have a sum of one, determine the weights of $\mathbf{x}_c$ and $\mathbf{x}_m$, respectively. Since $\mathbf{L}_m$ has a jagged shape, which does not match the real geological structure of faults, to achieve a balance between the accuracy and smoothness of faults, we commonly set λ_c slightly greater than λ_m; for example, $\lambda_c = 0.6$, and $\lambda_m = 0.4$. In addition, to filter out high-frequency components, we apply a smoothing filter on $\mathbf{x}_s$ and obtain the optimized labeling, $\mathbf{L}_s = (\hat{\mathbf{x}}_s, \mathbf{z}_c)$, depicted as the red curve in Fig. 3.9d. Finally, we connect the remaining lines in Figure 3.9a with the curve in Figure 3.9d, which is more geologically reasonable, and yield the labeled fault, as Figure 3.9e shows.

3.2.3 Fault Tracking Through Predicted Sections

Although the detection method introduced above shows accurate fault labeling results, it requires the tuning of several parameters, such as thresholds T_H, T_L, and T_A and weights λ_c and λ_m. To further improve interpretation efficiency, we employ the detection method only in some reference sections and track the detected faults through the remaining sections,

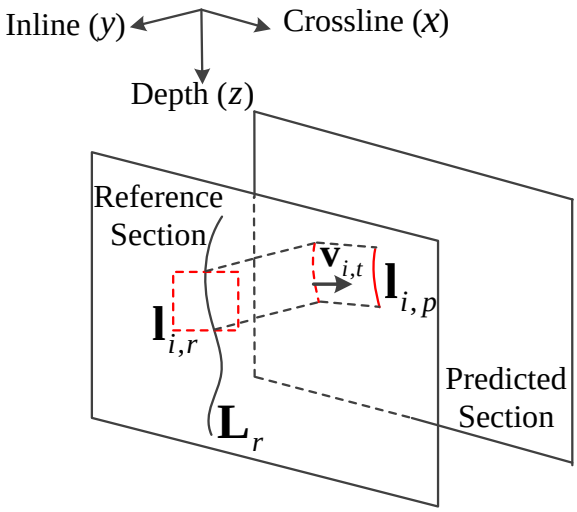


Figure 3.10: The process of tracking-vector-based projection.

referred to as "predicted sections." As Figure 3.5 shows, the tracking method contains two main parts, tracking-vector-based projection and tracked fault synthesis.

Tracking-Vector-Based Projection

In the tracking process, we project detected faults onto the most suitable positions in predicted sections, which utilizes fault information in reference sections. These projected positions are indicated by tracking vectors, the concept of which is similar to motion vectors in video coding. Figure 3.10 illustrates the process of the tracking-vector-based projection, in which x, y, and z correspond to the crossline, inline, and depth directions, respectively. In one reference section, we denote detected fault $\mathbf{L}_r = (\mathbf{x}_r, \mathbf{z}_r)$, in which $\mathbf{x}_r$ and $\mathbf{z}_r$ contain the x and z coordinates of all points. Fault segments, denoted $\mathbf{l}_{i,r}$, $i = 1, 2, \cdots$, are the subsets of $\mathbf{L}_r$ and identified by an analysis window with length r_t moving along the detected fault. In the piecemeal projection, we project segment $\mathbf{l}_{i,r}$ onto a target predicted section along the inline direction and keep all coordinates of $\mathbf{l}_{i,r}$ unchanged in the predicted section. In Figure 3.10, a fault segment in a dashed analysis window is projected onto the predicted section as a dashed red curve. Since fault surfaces are commonly not parallel to the inline direction, the initial projection is not accurate enough to represent the fault in the

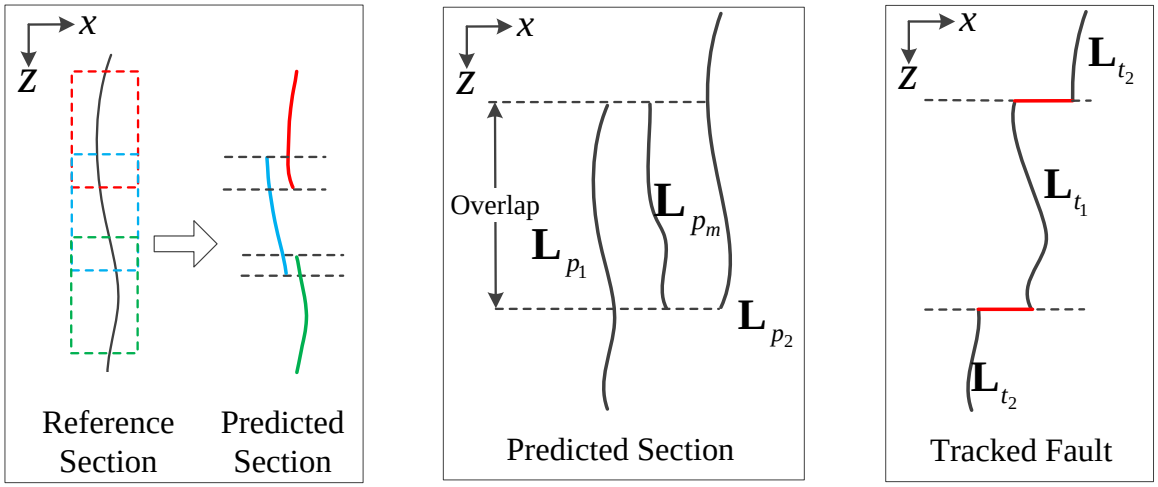


Figure 3.11: (a) The illustration of overlaps in the crossline direction, (b) projected faults $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ and line $\mathbf{L}_{p_m}$ with the largest discontinuity values, and (c) the synthesized fault based on fault tracking.

predicted section. Therefore, to identify the accurate positions of faults, we search for the optimal tracking vector in different directions under the constraint of maximizing the total discontinuity value of all points in $\mathbf{l}_{i,r}$. The objective function, which determines tracking vector $\mathbf{v}_i$, is

$$\mathbf{v}_i = \arg\max_{\mathbf{v}} \sum_{n=1}^{|\mathbf{l}_{i,r}|} D_p(\mathbf{l}_{i,r}(n) + \mathbf{v}), \tag{3.14}$$

where $|\mathbf{l}_{i,r}|$ defines the length of $\mathbf{l}_{i,r}$, D_p corresponds to the discontinuity map of the predicted section, and $\mathbf{l}_{i,r}(n) = (\mathbf{x}_{i,r}(n), \mathbf{z}_{i,r}(n))$ represents the coordinates of the n-th point in $\mathbf{l}_{i,r}$. Finally, obtained tracking vector $\mathbf{v}_i$ moves all points in $\mathbf{l}_{i,r}$ to new positions, as shown in Eq.(3.15):

$$\mathbf{l}_{i,p}(n) = \mathbf{l}_{i,r}(n) + \mathbf{v}_i, \ n = 1, 2, \cdots, |\mathbf{l}_{i,r}|\,. \tag{3.15}$$

In Figure 3.10, projected fault segment $\mathbf{l}_{i,p}$ is labeled by the solid red line in the predicted section. As we mentioned above, the moving of the analysis window identifies the fault segments of the detected faults. Since the moving step is smaller than the length of these segments, overlapping between neighboring fault segments in reference sections may prop-

agate to the predicted section. As dashed lines in Figure 3.11a depict, the projections of three neighboring fault segments overlap in the crossline direction. To synthesize more reliable projected faults, we average the x-coordinates of these overlaps and connect the averaged results with the remaining non-overlapped projections. In addition, since the projection is bi-directional, every fault in the predicted section can be delineated based on projected faults from the two neighboring reference sections. In Figure 3.11b, we define two projected faults as $\mathbf{L}_{p_i} = (\mathbf{x}_{p_i}, \mathbf{z}_{p_i})$, $i = 1, 2$, which will be involved in the synthesis of tracked faults.

Tracked Fault Synthesis

Although projected faults, $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, contain fault information propagated from reference sections, to synthesis tracked faults with high accuracy, we need to employ discontinuity information by defining another line, $\mathbf{L}_{p_m} = (\mathbf{x}_{p_m}, \mathbf{z}_{p_m})$, in the predicted section. Every point of $\mathbf{L}_{p_m}$ has the largest discontinuity value in the crossline interval between $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$. As Figure 3.11b shows, the length of $\mathbf{L}_{p_m}$ is determined by the overlapped region of $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ in the depth direction, referred to as $|\mathbf{z}_{p_m}| = |\mathbf{z}_{p_1} \cap \mathbf{z}_{p_2}|$, where $|\cdot|$ indicates the number of points in the overlap of projected faults. For the n-th point in $\mathbf{z}_{p_m}$, we obtain its corresponding x-coordinate by searching between $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ as follows:

$$\mathbf{x}_{p_m}(n) = \underset{x \in [x_{p_1}, x_{p_2}]}{\arg\max} D_p\left(x, \mathbf{z}_{p_m}(n)\right), n = 1, 2, \cdots, |\mathbf{z}_{p_m}|, \tag{3.16}$$

where x_{p_1} and x_{p_2} represent the x-coordinates of $\mathbf{z}_{p_m}(n)$ in $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, respectively.

On the basis of lines $\mathbf{L}_{p_1}$, $\mathbf{L}_{p_2}$, and $\mathbf{L}_{p_m}$, we attempt to synthesize tracked faults in predicted sections. Since $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ have different lengths in the depth direction, we divide the tracked fault into two parts. One is synthesized from the overlapped regions of $\mathbf{L}_{p_1}$, $\mathbf{L}_{p_2}$, and $\mathbf{L}_{p_m}$, as depicted in Figire 3.11b, and the other contains the remaining parts of $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$. We define the first part of the tracked fault as $\mathbf{L}_{t_1}$=$(\mathbf{x}_{t_1}, \mathbf{z}_{p_m})$, in which we

obtain $\mathbf{x}_{t_1}$ as follows:

$$\begin{aligned}\mathbf{x}_{t_1} = \arg\min_{\mathbf{x}} \mu_p \cdot \left(\lambda_{p_1} \cdot \left\|\mathbf{x} - \mathbf{x}'_{p_1}\right\|_2^2 + \lambda_{p_2} \cdot \left\|\mathbf{x} - \mathbf{x}'_{p_2}\right\|_2^2\right) \\ + \mu_m \left\|\mathbf{x} - \mathbf{x}_{p_m}\right\|_2^2 + \delta \left\|\mathbf{x}\right\|_2^2,\end{aligned} \tag{3.17}$$

where $\mathbf{x}'_{p_1}$ and $\mathbf{x}'_{p_2}$ represent the x-coordinates of $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$ in the overlapped regions, respectively. $\|\mathbf{x}\|_2^2$ is the normalization item, and $\sqrt{\delta}$ is the ratio of the minimum length of tracking vectors and the largest x-coordinate. μ_p and μ_m, the sum of which equals one, correspond to the weights of the projected faults and $\mathbf{L}_{p_m}$. Since we mainly depend on the projected faults in the predicted sections, μ_p is slightly greater than μ_m; for example $\mu_p = 0.6$ and $\mu_m = 0.4$. When the tracking process is applied on predicted sections, which correspond to same reference sections, parameters μ_p, μ_m, and δ remain unchanged. λ_{p_1} and λ_{p_2}, the weights of two projected faults, are determined as

$$\lambda_{p_i} = 1 - \frac{N_i}{N_1 + N_2}, \quad i = 1, 2 \quad , \tag{3.18}$$

where N_1 and N_2 represent the difference of inline coordinates between the predicted section and the two reference sections. We notice that λ_{p_i} negatively correlates to N_i, which means that the reference section closer to the predicted section corresponds to a higher weight, as the objective function in Eq. (3.17) shows. Synthesized part $\mathbf{L}_{t_1}$ is illustrated in the overlapped region of Fig. 3.11c. Finally, we connect the remaining parts of $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, denoted $\mathbf{L}_{t_2}$, with $\mathbf{L}_{t_1}$, which yields the tracked fault, as shown in Figure 3.11c.

3.2.4 Fault Similarity Index Measurement

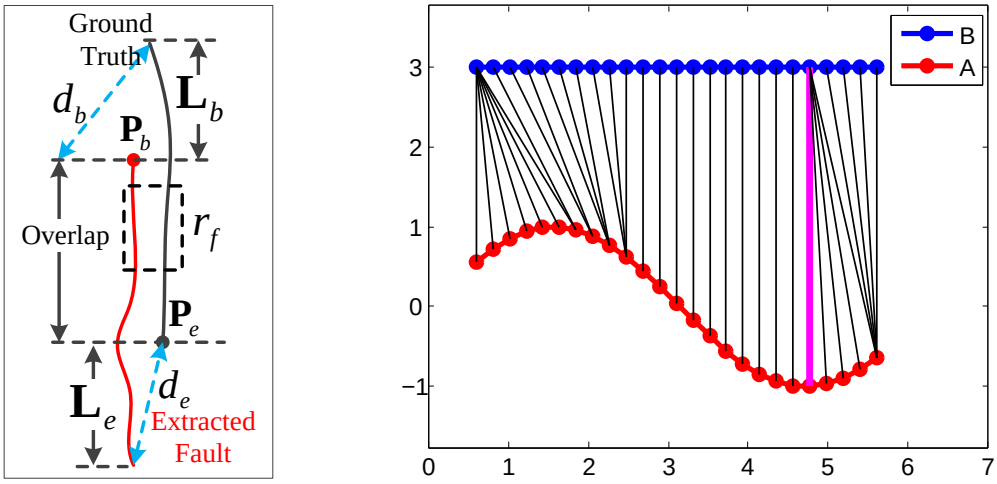
By employing the interactive fault interpretation method based on the 2D Hough transform and tracking vectors, we can semi-automatically delineate faults in the seismic sections of 3D seismic data. To evaluate the performance of this method, we introduce a fault similarity (FauSIM) index that measures the similarity between semi-automatically extracted

results and faults picked by experienced interpreters. Figure 3.12a illustrates an example of an extracted fault and its corresponding manually picked fault. The FauSIM index is developed based on the Fréchet distance [114], which can more accurately measure the similarity between two curves than the Haudorff distance [115] by taking the continuity of curves and the ordering of sampling points into account. To intuitively understand the Fréchet distance, we assume a situation, in which a man and his dog walking on two different paths vary their speeds but do not walk backwards. Under these conditions, the Fréchet distance between the two curves is the minimum length of a leash necessary when the man and the dog move along two separate curves from the starting points to the end points. Typically we calculate the Fréchet distance in Euclidean space S. Curves A and B in S are defined by two continuous mappings, $A : [0, J] \rightarrow S$, and $B : [0, K] \rightarrow S$, which map walking distances to vectors in S. J and K represent the length of the two curves, respectively. In addition, we define another two continuous and non-decreasing functions within the interval of $[0, 1]$, $\alpha(t)$ and $\beta(t)$, where $\alpha(0) = 0$, $\alpha(1) = J$, $\beta(0) = 0$, and $\beta(1) = K$. Both functions map normalized walking time t to the walking distance in each path, respectively. Therefore, based on these pre-defined functions, the Fréchet distance $F(A, B)$ between curves A and B can be calculated as

$$SIM(A, B) = \inf_{\alpha,\beta} \left\{ \max_{t\in[0,1]} d\left(A\left(\alpha(t)\right), B(\beta(t))\right) \right\}, \tag{3.19}$$

where $d(\cdot)$ is the distance function defined in S, referred to as $\|\cdot\|_2$. Figure 3.12b illustrates an example of the Fréchet distance, where black lines connect matched points in curves A and B, and the Fréchet distance is the length of the magenta line, which corresponds to the longest connection.

According to the defined Fréchet distance, we attempt to numerically describe local and global similarity between labeling results and the ground truth using local and global items in the FauSIM index, respectively. Since in most cases two curves have different



(a) (b)

Figure 3.12: (a) the comparison of an extracted fault and its ground truth labeled by interpreters, and (b) The Fréchet distance between curves A and B can be represented by the length of the magenta line.

lengths, we focus on comparing the local similarity of only overlapped regions in the depth direction and evaluate the similarity of remaining parts in the global item of the FauSIM index. To compare the local details of an extracted fault and a manually picked fault, for any z-coordinate in the overlapped region, we define an analysis window with length r_f that identifies a pair of line segments, as Fig. 3.12a shows. Therefore, by moving the analysis window along the overlapped region and calculating the Fréchet distance of every pair of local segments, we can obtain a sequence of distances, $\mathbf{d} = (d_i)_{i=1}^{N_d}$, where N_d is the number of the pairs. To increase the reliability of the local item, we employ discontinuity values as the weights of obtained distances in $\mathbf{d}$. For each pair of local segments, the corresponding weight, denoted $\bar{D}_i$, $i = 1, 2, \cdots, N_d$, is defined as the average discontinuity value of the segment extracted by the moving of the analysis window along the extracted fault. Therefore, the weighted mean and standard deviation of $\mathbf{d}$ can be calculated as

$$\mu_d = \frac{\sum_{i=1}^{N_d} e^{-\bar{D}_i} d_i}{\sum_{i=1}^{N_d} e^{-\bar{D}_i}}, \qquad \sigma_d = \sqrt{\frac{\sum_{i=1}^{N_d} e^{-\bar{D}_i} (d_i - \mu_d)^2}{\sum_{i=1}^{N_d} e^{-\bar{D}_i}}}. \tag{3.20}$$

In Eq. (3.20), we use $\exp(-\bar{D}_i)$ instead of $\bar{D}_i$ as weights in order to add penalties to the local segments of extracted faults, which are not located around incoherent regions. For two pairs of local segments with the same Fréchat distance, the one with smaller $\bar{D}_i$ corresponds to a relatively greater weighted distance in Eq. (3.20). Therefore, greater μ_d and σ_d represent lower similarity between extracted faults and manually picked faults. In the FauSIM index, μ_d and σ_d constitute the main part of the local item.

In addition to the comparisons of local segments in overlapped regions, we also need to consider the influences of remaining parts, denoted $\mathbf{L}_b$ and $\mathbf{L}_e$ as Figure 3.12a shows, on the similarity of extracted faults to manually picked faults. The appearance of $\mathbf{L}_b$ and $\mathbf{L}_e$ results from the different positions of the starting and ending points of two compared curves. To employ the influences of $\mathbf{L}_b$ and $\mathbf{L}_e$ in the FauSIM index, we define d_b and d_e illustrated in Fig. 3.12a as the lengths of lines connecting the starting and ending points of curves, respectively. From another perspective, d_b can also be regarded as the Fréchet distance between $\mathbf{L}_b$ and the starting point with a smaller z-coordinate, denoted $\mathbf{P}_b$ in Figure 3.12a. Similarly, d_e represents the Fréchet distance between $\mathbf{L}_e$ and the ending point with a larger z-coordinate, denoted $\mathbf{P}_e$ in Figure 3.12a. In addition, we define d_{max} as the Fréchet distance of two entire curves. Therefore, d_b, d_e, and d_{max} form the main part of the global item in the FauSIM index.

Based on obtained parameters μ_d, σ_d, d_b, d_e, and d_{max}, we define the FauSIM index, which numerically characterizes local and global similarity between labeling results and manually picked faults using local and global items. The expression of the FauSIM index is shown as follows:

$$FauSIM = \underbrace{\exp\left(-\alpha(\mu_d + \sigma_d)\right)}_{\text{Local Item}} \cdot \underbrace{\exp\left(-\beta\left(\frac{d_{max} + c_b d_b + c_e d_e}{1 + c_b + c_e}\right)\right)}_{\text{Global Item}}, \tag{3.21}$$

where α and β are normalization items determined empirically for local and global items, respectively, and c_b and c_e represent the weights of d_b and d_e, respectively. Since function

$\exp(\cdot)$ defined on negative real numbers has a range of 0 to 1, we apply it in the FauSIM index for normalization. c_b and c_e are defined as the ratios of the lengths of $\mathbf{L}_b$ and $\mathbf{L}_e$ and the length of the extracted fault, respectively. Therefore, according to the expression of FauSIM in Eq. (3.21), the greater FauSIM value represents higher similarity between the semi-automatically labeled fault and the manually picked fault and vice versa.

3.3 Experimental Results

To verify the reliability and robustness of the interactive fault detection and tracking methods, in this section, we apply this interpretation method to detect faults in real seismic datasets and involve the FauSIM index as an objective measure. The selected 3D seismic dataset, acquired from the $16 \times 24\,\text{km}^2$ Netherland offshore F3 block in the North Sea, contains typical geological structures such as faults and salt domes [116]. In this dataset, the inline number ranges from #100 to #750, and the crossline number ranges from #300 to #1250. In addition, in the time direction, the dataset has a range of 0 ms to 1848 ms in 4ms steps. Therefore, the 3D dataset has a size of $651 \times 951 \times 463$. In the following examples, we focus on two separate local volumes that contain different faults. One has a major fault illustrated in Figure 3.6a. The other contains multiple small faults, and Figure 3.13 illustrates one of its seismic sections.

3.3.1 Fault Detection in Reference Sections

The first step of detecting faults in seismic sections is to apply threshold $T_H = 0.9$ on the discontinuity map of the seismic section. The obtained binary image, shown in Figure 3.6a, highlights likely fault regions. Furthermore, we extract fault features from the highlighted likely fault regions using the Hough transform and label them as yellow lines in the magnified seismic section shown in Figure 3.14a. The red line in Figure 3.14a, which represents a fitted line, helps define the *lateral distance* and identify *outliers*. By selecting thresholds $T_L = T_A = 5$, we remove *outliers* and merge *neighboring groups*. In Figure 3.14b, the

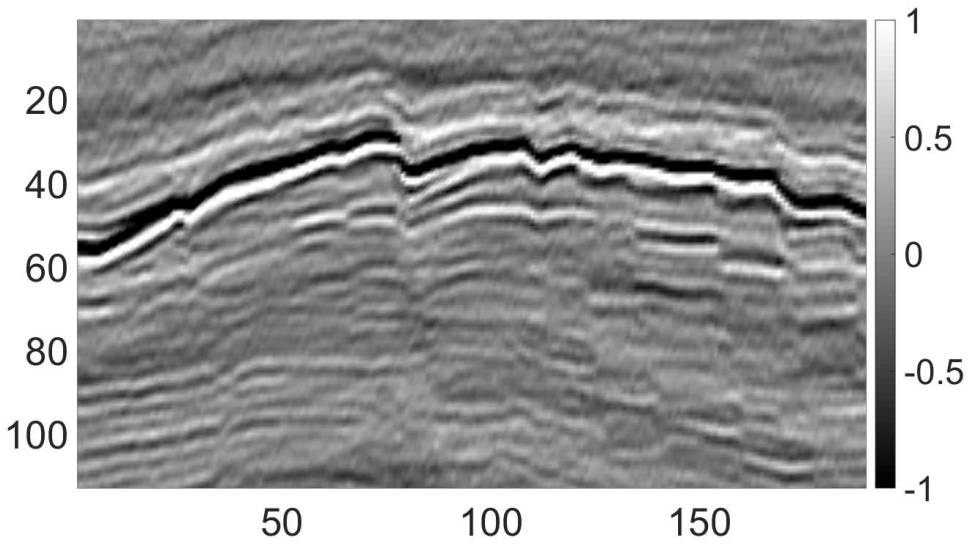
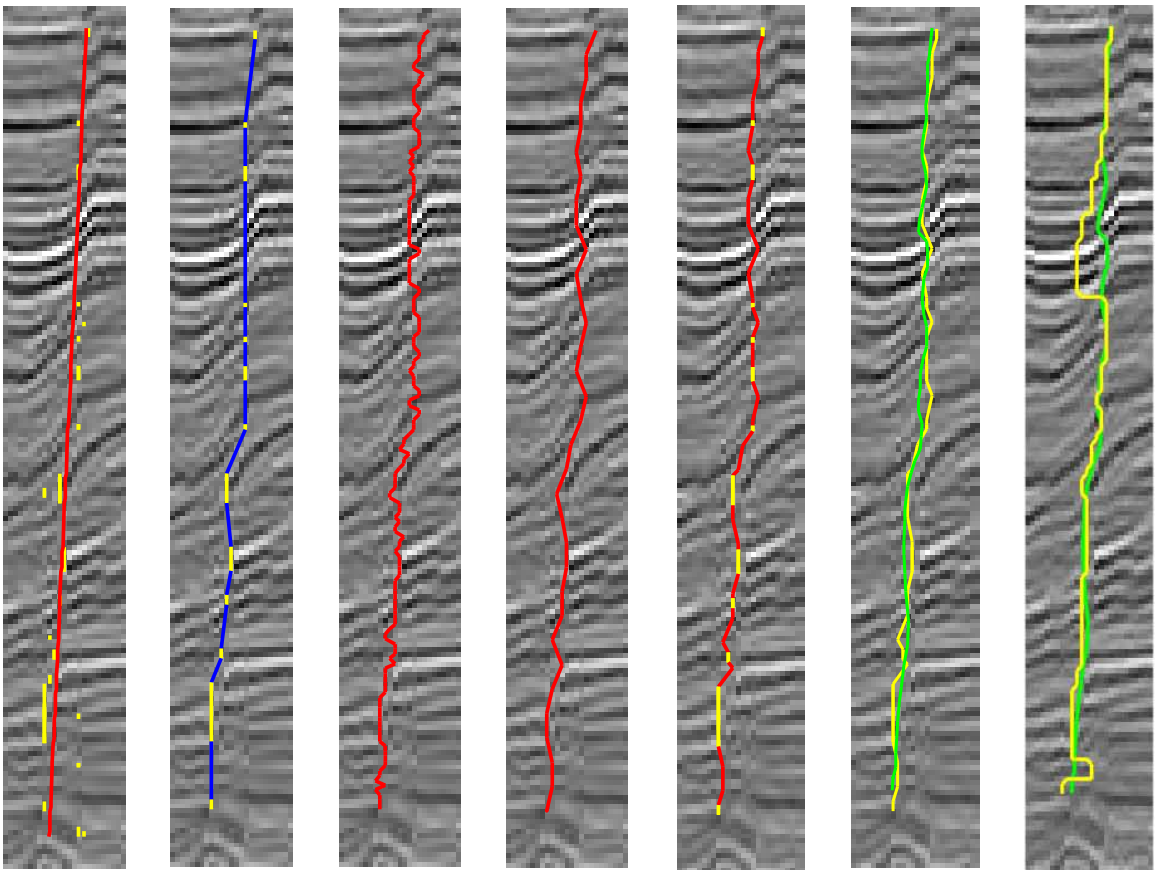


Figure 3.13: The seismic section of a local volume that contains multiple faults

remaining features, connected by blue lines, form the initial labeling of a fault, denoted as $\mathbf{L}_c$ in the previous section. Since the connection of neighboring features does not involve any geological constraint, to obtain more accurate detection result, we search horizontally within radius $r_s = 2$ and identify locations with the largest discontinuity values. Figure 3.14c illustrates the searching result, $\mathbf{L}_m$. On the basis of Eq. (3.13), we combine $\mathbf{L}_c$ and $\mathbf{L}_m$ with the corresponding weights $\lambda_c = 0.6$ and $\lambda_m = 0.4$, respectively. After applying a smoothing filter on the combined result, we obtain the optimized labeling shown in Figure 3.14d. Finally, by connecting the remaining features with the optimized labeling, we delineate the fault, shown in Figure 3.14e.

To evaluate the accuracy of the fault detection method based on the Hough transform, we compare the detected fault in Figure 3.14f with a manually picked fault. From the subjective point of view, we notice the strong similarity between the detected fault and the manually picked fault. To compare the performance of the detection method, we implement an efficient fault detection method proposed by D. Hale in [66] and include its detected results as a benchmark. Hale's method first smooths along fault orientations and delineates faults by identifying points with the largest discontinuity values on the horizontal direction. However, without involving geological constraints in the labeling process, the appearance



(a) (b) (c) (d) (e) (f) (g)

Figure 3.14: The process of fault labeling, (a) extracted fault features and the fitted line, (b) remaining features after false feature removal form the initial labeling, (c) searching results based on the discontinuity map, (d) the optimized labeling, (e) the detected fault, (f) the fault (yellow) detected by our fault detection method and the manually picked fault (green), and (g) the fault (yellow) labeled by Hale's method [66] and the manually picked fault (green).

of outliers in Figure 3.14g degrades the similarity between the detected fault and the manually picked fault. From the objective point of view, Table 3.1 shows the FauSIM indices of detected faults. In the inline section containing only a single fault (#268), the fault detected by our method has a FauSIM index greater than that detected by Hale's method, which complies with our subjective impression.

The example in Figure 3.14 shows that the detection method based on the Hough transform has capability of detecting faults in seismic sections that contain simple faults. In addition, such a detection method can also label multiple faults in seismic sections. The

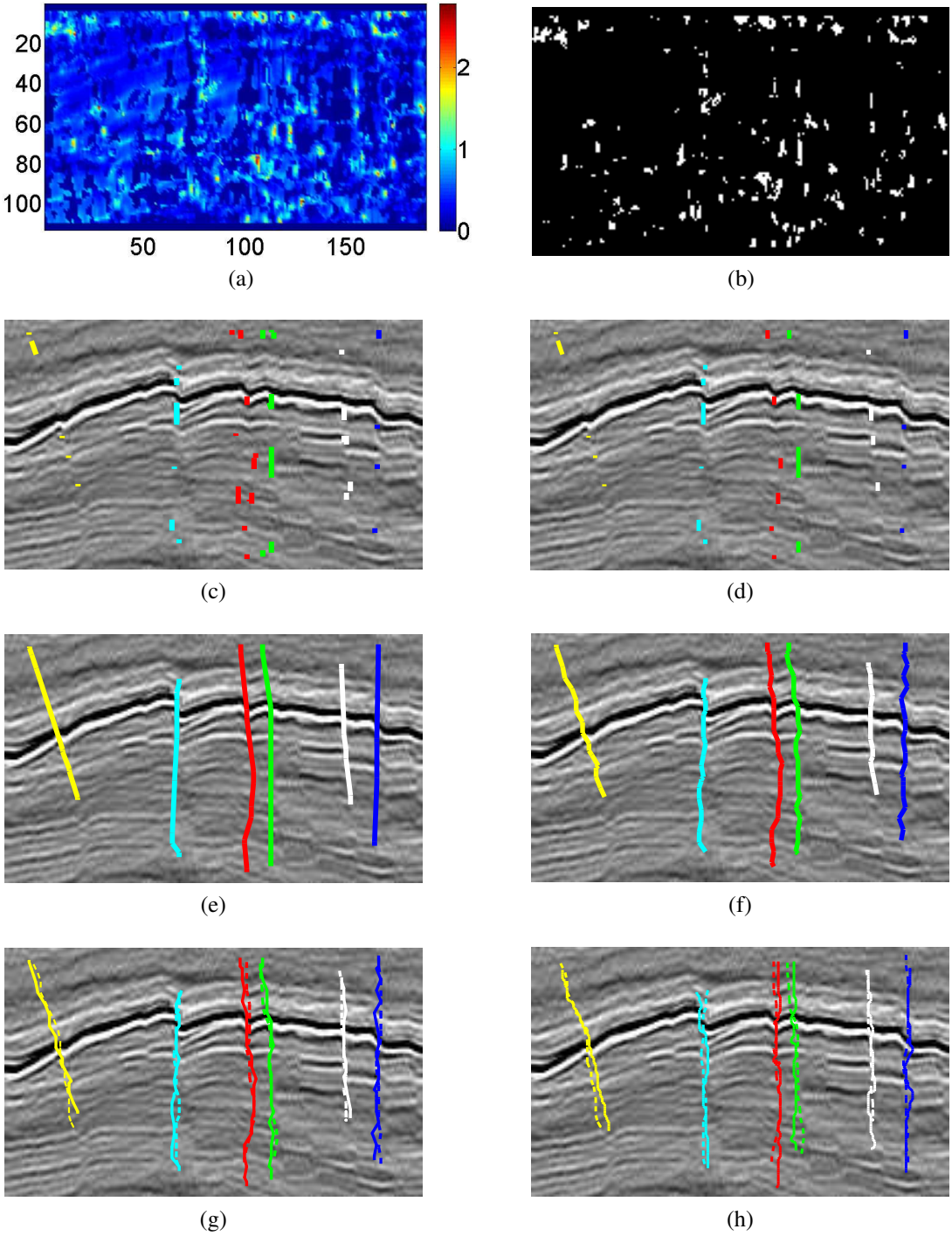
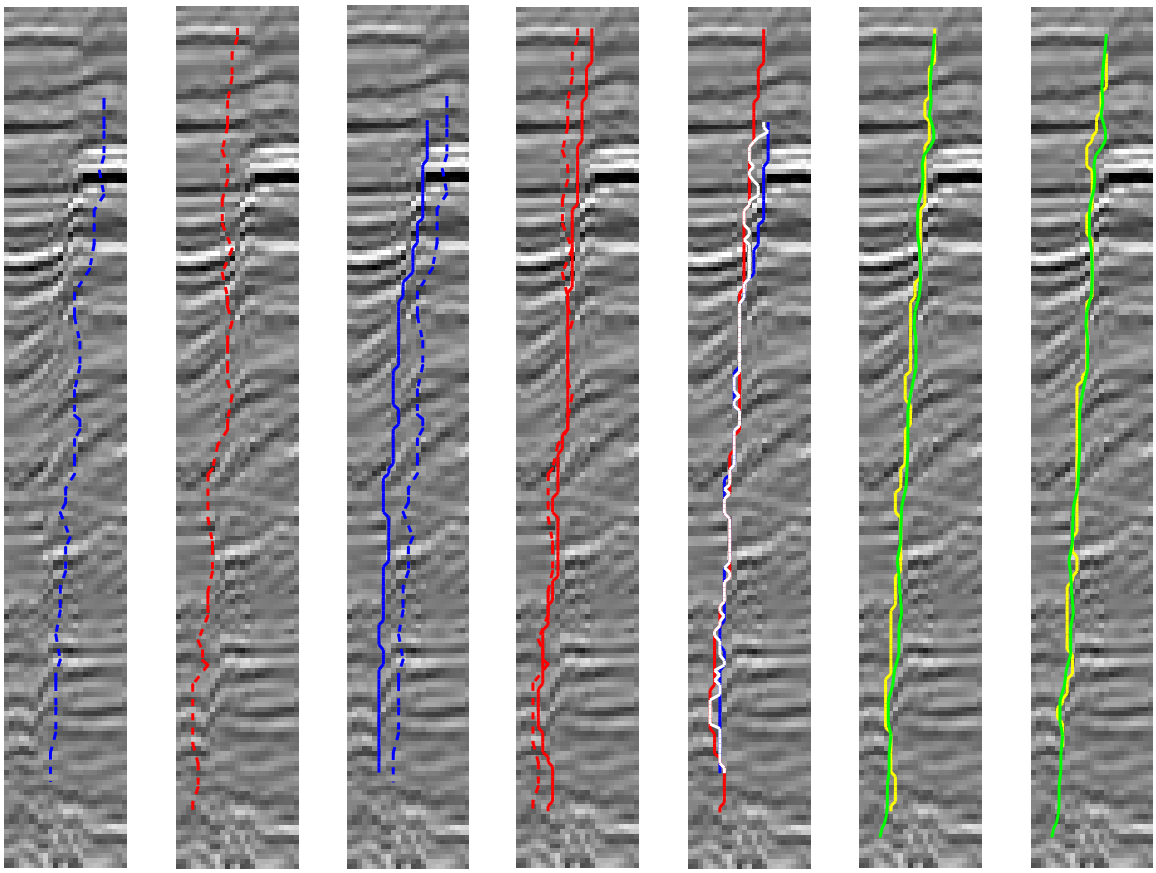


Figure 3.15: The detection of multiple faults, (a) the discontinuity map of Inline #244, (b) highlighted likely fault regions, (c) detected features in different groups labeled by different colors, (d) remaining features after removing false features, (e) the initial labeling of faults, (f) detected faults with geological constraints involved, (g) the comparison between faults detected by our detection method and the manually picked fault, and (h) the comparison between faults detected by Hale's method and the manually picked fault.

Table 3.1: The FauSIM indices of faults detected by different detection methods in seismic sections

Inline Section	Our Method	Hale's Method [66]
#268 (Single Fault)	**0.8487**	0.7402
#244 (Multiple Faults)	**0.8127**	0.7719

only difference is to divide detected features into groups, the number of which is determined by interpreters and coincides with the number of faults. Figure 3.15a shows the discontinuity map of Inline #244, and Figure 3.15b illustrates likely fault regions extracted from Figure 3.15a. By observing the seismic section in Figure 3.13, we distinguish six apparent faults. Therefore, on the basis of the positions and slopes of faults, we apply *k*-means clustering method on the midpoints of detected fault features to classify detected features in Figure 3.15c into six groups and label them by different colors. After removing false features, we obtain the remaining features of each group, as Fig. 3.15d illustrates. In Figure 3.15e, the line connection of neighboring features in each group forms the initial labeling of faults. In the end, by involving the discontinuity information, we detect faults with high accuracy as Figure 3.15f shows. In Figure 3.15g, we compare the detected faults with manually picked faults in dashed lines and subjectively conclude that the detected results closely resemble manually picked faults. In contrast, Figure 3.15h illustrates the comparison between manually picked faults in dashed lines and faults detected by Hale's method, in which the main difference comes from the deviations of yellow and green faults. To objectively compare the performance of our and Hale's methods, in Table 3.1, we list the FauSIM indices of faults detected in different seismic sections. For the seismic section containing multiple faults, the corresponding FauSIM index is the mean of the similarity indices of all detected faults. We notice that our method has greater FauSIM indices than Hale's method, which is consistent with our subjective assessment. Parameters involved in the process of fault detection are T_H, T_L, T_A, r_s, λ_c, and λ_m, all of which are determined empirically. However, because of the similar structures of faults in one dataset, users do



(a) (b) (c) (d) (e) (f) (g)

Figure 3.16: The synthesis of the tracked fault in Inline #258, (a) the reference fault detected in Inline #248, (b) the reference fault detected in Inline #268, (c) the projected fault, $\mathbf{L}_{p_1}$, in solid curve, (d) the projected fault, $\mathbf{L}_{p_2}$, in solid curve, (e) $\mathbf{L}_{p_1}$, $\mathbf{L}_{p_2}$, and $\mathbf{L}_{p_m}$ for the synthesis of the tracked fault, (f) the comparison between the tracked fault (yellow) and the manually picked fault (green), and (g) the comparison between the detected fault (yellow) and the manually picked fault (green).

not need to change or only need to slightly change the values of certain parameters when detecting faults in different seismic sections.

3.3.2 Fault Tracking Through Predicted Sections

If we depend only on the detection method, the tweaking of parameters in each section may reduce interpretation efficiency. Therefore, on the basis of detected faults in reference sections, we conduct the fault tracking method to label faults in predicted sections. In the local volume containing the single fault, we detect faults using the detection method

in two reference sections, Inline #248 and Inline #268. To synthesize the tracked fault in Inline #258, we project faults detected in reference sections onto the target section and keep the positions of faults unchanged. As Figures 3.16a and 3.16b illustrate, in magnified Inline #258, dashed curves represent the detected faults of reference sections. By moving an analysis window with length $r_t = 30$ along each reference fault, we obtain a group of fault segments, which overlap on the depth direction because of smaller moving step $r_m = 5$. According to the constraint in Eq. (3.14), for each fault segment, we search in different directions to identify the corresponding tracking vector, which shifts the fault segment to the most discontinuous position. After merging all shifted fault segments, we obtain projected faults, $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, as the solid curves shown in Figures 3.16c and 3.16d, respectively. In the synthesis of tracked faults, we focus on the overlaps of projected faults on the depth direction. To increase the accuracy of the tracked fault, we define line $\mathbf{L}_{p_m}$ between $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, every point of which corresponds to the largest discontinuity value on the crossline direction. The white curve in Figure 3.16e represents $\mathbf{L}_{p_m}$ obtained in Inline #258. On the basis of $\mathbf{L}_{p_1}$, $\mathbf{L}_{p_2}$, and $\mathbf{L}_{p_m}$, we can synthesize the tracked fault according to the objective function in Eq. (3.17). In this function, parameters λ_{p_i}, $i = 1, 2$, depend on the distance between the predicted section (Inline # 258) and two reference sections (Inline #248 and #268). Therefore, in this case, $\lambda_{p_1} = \lambda_{p_2} = 0.5$. On the other hand, parameters μ_p and μ_m are determined empirically. To reduce the influence of the zig-zag shape of $\mathbf{L}_{p_m}$, we choose $\mu_p = 0.6$ and $\mu_m = 0.4$. In Figure 3.16f, we label the tracked fault in yellow and compare it with the manually picked fault in green. Because of involving fault information from two reference sections, the tracked fault in Figure 3.16f shows strong similarity to the manually picked fault. In contrast, we extract the fault depending only on the detection method and compare it with with the manually picked fault in Figure 3.16g. We notice that the most parts of the detected fault are similar to the manually picked fault, except for the major deviation in the upper area. To objectively compare faults labeled in Figures 3.16f and 3.16g, we list the corresponding FauSIM indices in Table 3.2. For Inline #259, the

Table 3.2: The comparison of FauSIM indices between tracked and detected faults in seismic sections

Inline Section	Tracked Faults	Detected Faults
#258 (Single Fault)	**0.8632**	0.8321
#249 (Multiple Faults)	**0.8094**	0.7611

tracked fault has greater similarity index than the detected one, which complies with our subjective observation. Since the detection method focuses only on one seismic section, ignoring the coherency between neighboring sections leads to a limitation of similarity.

As we introduced above, Figure 3.16 illustrates the process of synthesizing tracked faults. To show the robustness of the tracking method, we apply it on a local volume containing multiple faults. In the synthesis of tracked faults, we first detect faults in selected reference sections (Inline #244 and Inline #254) and label them by dashed curves depicted in Figs. 3.17a and 3.17b, respectively. Different colors distinguish different faults. For each detected fault, we define a group of fault segments and search in different directions to determine the most discontinuous projected positions. After merging shifted fault segments, we yield projected faults, $\mathbf{L}_{p_1}$ and $\mathbf{L}_{p_2}$, and label them by solid curves as Figs. 3.17c and 3.17d illustrate. We notice that faults projected from Inline #244 are different from those projected from Inline #254. The main reason is the termination of faults in predicted sections. Therefore, in the tracking process, we focus only on faults appearing in both reference sections such as faults labeled by cyan, red, green, white, and blue in Figure 3.17. For a fault detected in only one reference section such as the yellow fault projected from Inline #244 and the magenta fault projected from Inline #254, interpreters need to decide whether this fault exists in the target predicted section by previous experience. In Figures 3.17e and 3.17f, we compare the manually picked fault in orange with tracked faults and detected faults, respectively. By only observing results shown in Figs. 3.17e and 3.17f, we can not easily decide which one is better, since detected faults are also similar to the manually picked fault, except for the difference in length. However, by calculating the

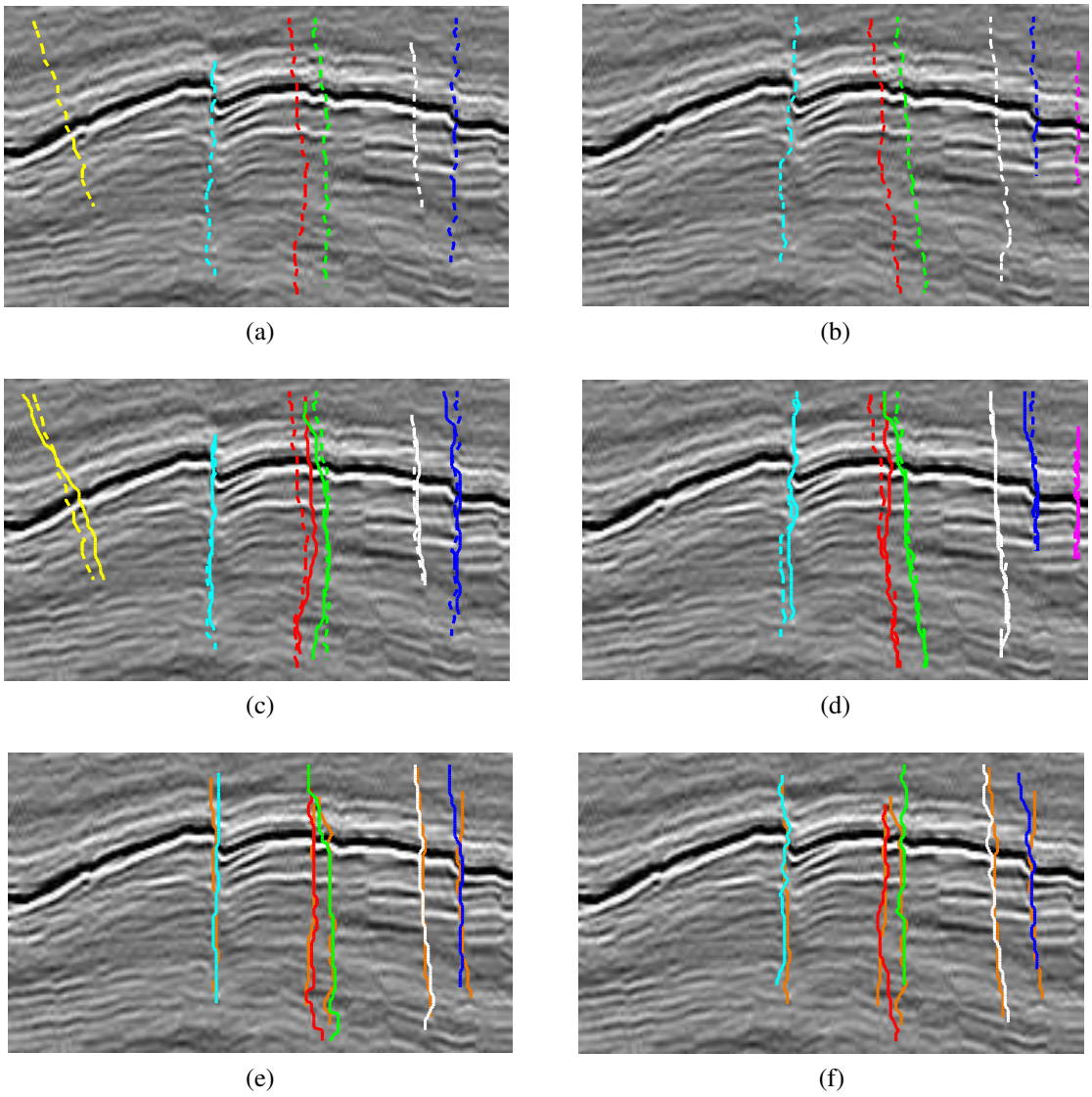


Figure 3.17: The synthesis of tracked faults in Inline #258, (a) reference faults detected in Inline #244, (b) reference faults detected in Inline #254, (c) projected faults, $\mathbf{L}_{p_1}$, in solid curves, (d) projected faults, $\mathbf{L}_{p_2}$, in solid curves, (e) the comparison between tracked faults in Inline #249 and the manually picked fault, and (d) the comparison between detected faults in Inline #249 and the manually picked fault.

FauSIM indices of tracked and detected faults, we can objectively evaluate the similarity to the manually picked fault. According to FauSIM indices shown in Table 3.2, we conclude that tracked faults have high similarity than detected faults, although in Fig. 3.17e the red and green faults overlap at the beginning area.

3.3.3 Overall Comparison

According to our previous experiments, we can extract fault surfaces using three different strategies. One is to apply Hale's method to detect faults in each seismic section, and another is to implement the fault detection of each seismic section using our detection method. The third strategy is a combined method that first detects faults in selected reference seismic sections and synthesizes tracked faults in predicted sections by involving fault information obtained from reference sections. To compare the performance of different strategies on labeling faults, we apply those methods to the local seismic volume containing the simple fault, the inline number of which ranges from 248 to 286 with step 2. In this volume, we select three reference sections (Inline # 248, #268, and # 286) and regard remaining ones as predicted sections. The 3D fault surface delineated by the combined method is shown in Fig. 3.18. In addition, Fig. 3.19 shows the FauSIM indices of faults delineated in each seismic section by different strategies, and Table 3.3 lists the corresponding means and standard deviations. Although we notice that in some seismic sections, faults obtained from our detection method are slightly more similar to manually picked faults than those detected by the combined method, means in Table 3.3 show that for this local volume the combined method has better performance. In contrast to the combined method, which employs information from multiple reference sections, our detection method has a lower standard deviation, which demonstrates its robustness on fault detection.

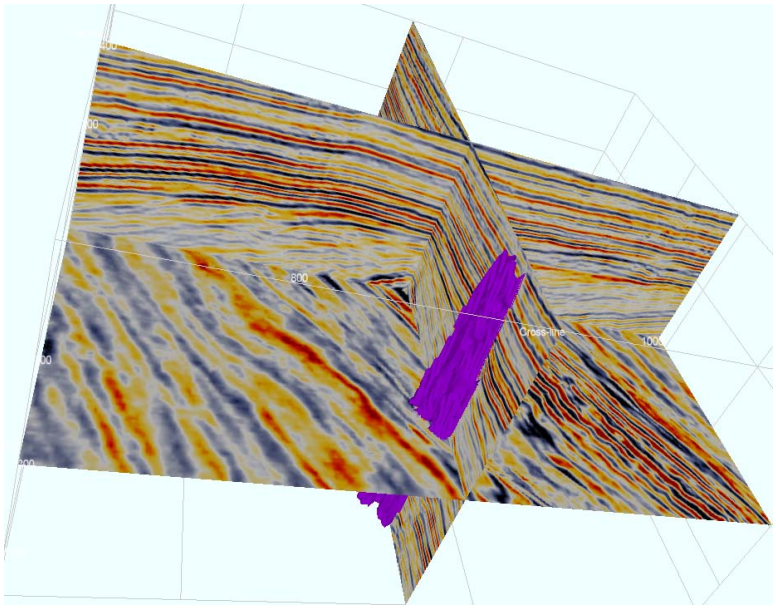


Figure 3.18: 3D fault surfaced delineated by the combined method.

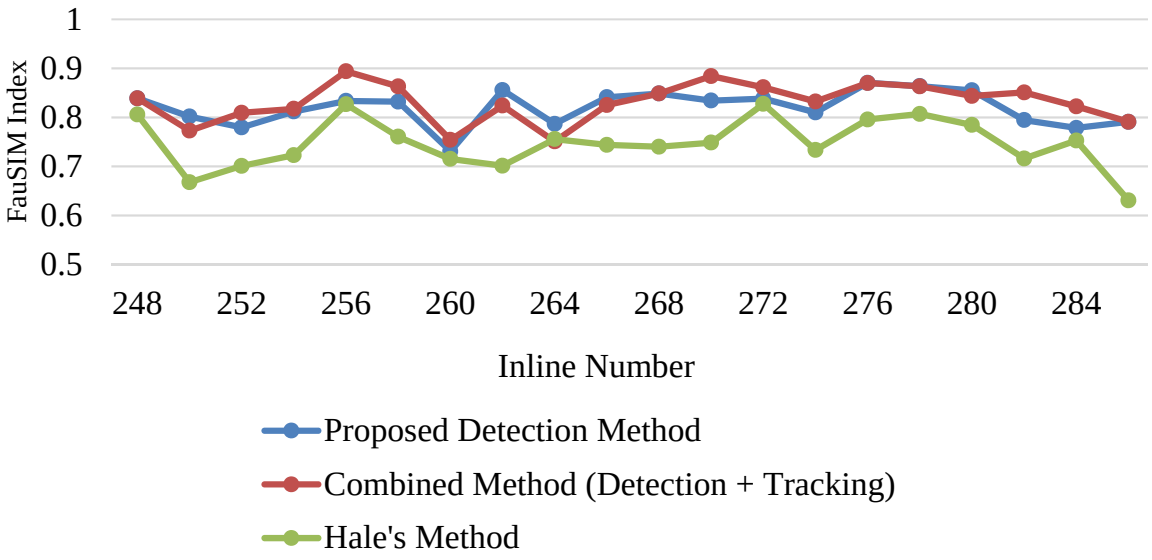


Figure 3.19: The FauSIM indices of faults in each seismic section (From Inline #248 to Inline #286) delineated by different detection methods.

Table 3.3: The means and standard deviations of FauSIM indices in Fig. 3.19

Different Methods	Mean	Std. Dev.
Our Detection Method	0.8197	**0.0355**
Combined (Detection + Tracking)	**0.8308**	0.0400
Hale's Method	0.7488	0.0512

3.4 Summary

In this chapter, we introduced an interactive interpretation framework that delineated faults in seismic volumes using their geological and geometric features. The geological feature of faults is their discontinuity along horizons. The geometric feature of faults is their line-like or curved shapes in seismic sections, which appear as curved surfaces in seismic volumes. To improve the efficiency of fault interpretation, we classified seismic sections into reference and predicted ones by borrowing the concept of I- and B-frames in video-coding techniques and applied different strategies on the two types of sections. In the reference sections, we utilized discontinuity maps to highlight discontinuous regions along horizons, referred to as likely fault regions, and employed the Hough transform to extract the line features of faults. Because of the appearance of noise and the limitation of the Hough transform, it was inevitable that detected results contain some false features that violate geological constraints. To remove false features, we characterized geological constraints using the spatial relationships of detected features. The connection of remaining features determines the accuracy of fault labeling. Simply connecting features using lines depicts a rough shape of a fault. However, such labeling is not accurate since lines connecting features involve no geological information. To improve interpretation accuracy, we searched horizontally along the initially labeled fault for the highest discontinuity value and obtained another candidate. The combination of these two candidates forms the optimal labelling of faults in reference sections. For predicted sections, we synthesized tracked faults using ones detected from two reference sections for high accuracy. We utilized the discontinuity attribute to estimate tracking vectors, which projected the local segments of faults labeled in reference sections to the corresponding position in the target predicted section. The optimal combination of fault segments projected from reference sections involves discontinuity and spatial constraints. Faults labeled in reference and predicted sections form the fault surface in the seismic volume. To evaluate the performance of the interactive framework on

fault interpretation, we introduced the fault similarity (FauSIM) index that describes the similarity between detected faults and manually picked faults. The FauSIM index based on the Fréchet distance compares both the local and global structures of faults. Experimental results showed that the interactive framework has the capability to accurately detect faults in seismic sections, and the objective FauSIM indices complied with the subjective observation of interpreters.

CHAPTER 4

TEXTURE ANALYSIS AND NOISE-ADJUSTED TENSOR-BASED INCREMENTAL LEARNING IN INTERACTIVE SALT-DOME INTERPRETATION

A salt dome as a dome-shaped structure is formed by the evaporation of a large mass of salt in sedimentary rocks. Salt domes that are impermeable structures prevent the migration of hydrocarbons and provide entrapment for oil and gas reservoirs. Figure 4.1 illustrates an example of a salt dome in one seismic section. We notice that the top boundary of a salt dome usually has a cap rock, which appears in migrated seismic data as areas with high contrast and strong edges. In contrast, the lateral boundaries in most geological scenarios lack such high contrast and are more difficult to detect because of physical constraints in the underlying imaging process. As Figure 4.1 shows, the lateral boundaries of the salt dome have no explicit edge existing between the salt region and the non-salt region containing horizons. However, we notice that chaotic reflections in a salt dome are in a form of a distinct texture, which is different from the non-salt region. Therefore, seismic interpreters are able to delineate such boundaries by observing the change of texture between salt and non-salt regions. To characterize the changes of textures, we will mainly focus on two types of texture attributes, gray-level-co-occurrence-matrix (GLCM)-based attributes and the gradient of texture (GoT) in this chapter.

As 3D geological structures in seismic volumes, salt domes can be interactively interpreted in a way similar to the labeling process of faults. We classify seismic sections into reference and predicted ones, where we introduce different strategies for delineating salt domes. We design the texture-attribute-based salt-dome detection workflow for reference sections and noise-adjusted tensor-based incremental principal component analysis (PCA) for tracking salt-dome boundaries through predicted sections. This chapter contains five

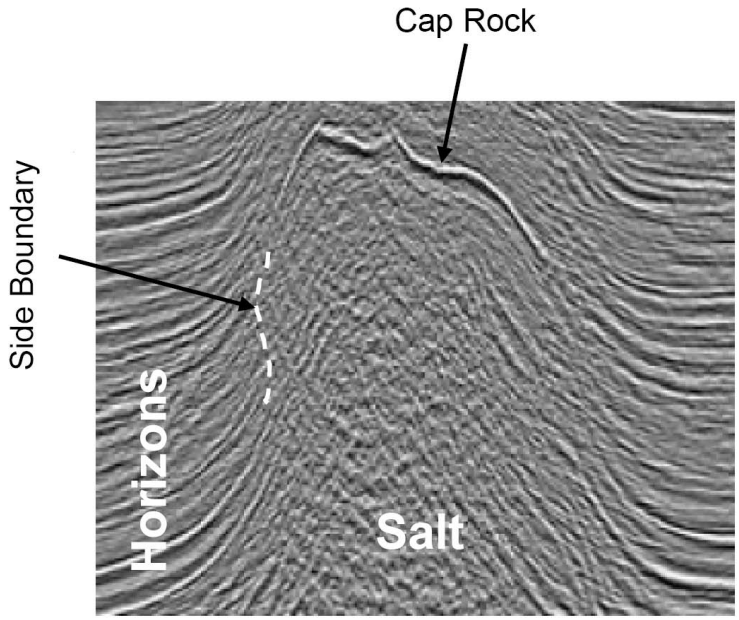


Figure 4.1: A seismic section contains a salt-dome, along the boundary of which local regions have different appearances.

sections. We first introduce texture attributes such as GLCM-based attributes and the GoT. Then we introduce the concept of tensors and multilinear algebra. Furthermore, we introduce the interactive salt-dome interpretation method and experimental results. Finally, we make a summary for this chapter.

Notations in this chapter follow conventions in the multilinear algebra [117]. We denote scalars using lowercase letters such as m and n, vectors using boldfaced lowercase letters such as $\mathbf{x}$ and $\mathbf{y}$, matrices using boldfaced uppercase letters such as $\mathbf{U}$ and $\mathbf{W}$, and tensors using boldfaced calligraphic letters such as $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$. In addition, we follow such notations when denoting the lower-order parts of a given structure. For example, the entry of matrix $\mathbf{A}$ with row index i and column index j is denoted as a_{ij}, and the ith column vector of matrix $\mathbf{A}$ is denoted as $\mathbf{a}_i$. To improve the overall readability, we make exceptions for the notations of scalars and pixels. Lowercase letters such as i, j, and k refer to indices or counters. In contrast, uppercase letters such as I, J, and K represent the upper bound of corresponding indices. In image $\mathbf{I}$, we denote the pixel located at i-th row and j-th column as $\mathbf{I}[i, j]$ by following the conventional notation in the domain of image processing.

4.1 Texture Attributes for Salt-dome Interpretation

4.1.1 Gray-level Co-occurrence Matrix (GLCM)

A GLCM [22] describes the distribution of co-occurring grayscale values at a given offset over an image. Given an $M \times N$ image with L grayscale levels, the entry of its corresponding GLCM is calculated as follows:

$$\mathbf{G}_{[\Delta x,\Delta y]}[i,j] = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\begin{cases}1, & \text{if } \mathbf{I}[x,y]=i \text{ and } \mathbf{I}\left[x+\Delta x, y+\Delta y\right]=j,\\ 0, & \text{otherwise,}\end{cases} \tag{4.1}$$

where $[x, y]$ indicates the spatial position in the image, $\mathbf{I}[x, y]$ represents the grayscale value of pixel $[x, y]$, and i and j as grayscale values range from 1 to L. Therefore, the number of grayscale levels determines the size of the GLCM. In addition, offset $[\Delta x, \Delta y]$ describes the spatial relationship between a pixel with grayscale value i and a pixel with grayscale value j. For an image, grayscale levels and one specific offset determine one of its GLCMs. Using Eq. (4.1), we obtain the GLCM of the entire image, from which global texture features can be extracted. However, in most cases we may need to analyze the texture features of local regions. To solve this problem, we calculate the GLCMs of local patches. Every pixel $[i, j]$ corresponds to an $N_p \times N_p$ analysis window, in which grayscale values are quantized into discrete levels. Since textures may have complicated details, features extracted from a GLCM with only one specific offset commonly fail to characterize all texture information. Therefore, by varying the directions and pixel distances of offset $[\Delta x, \Delta y]$, we obtain a series of GLCMs, which describe texture from different perspectives. The typical offset directions are $\{0°, 45°, 90°, 135°\}$, and the pixel distances of offsets are determined by the size of images or patches.

Statistical properties derived from GLCMs as scalars are able to describe texture features in different ways. Given N_g offsets, the popular statistical properties of GLCMs are listed as follows.

1. `Contrast` represents the contrast between pixels with various offsets and has an expression of

$$\texttt{Contrast} = \frac{1}{N_g} \sum_{[\Delta x, \Delta y]} \left(\sum_{i=1}^{L} \sum_{j=1}^{L} (i-j)^2 \mathbf{G}_{[\Delta x, \Delta y]}[i,j] \right). \tag{4.2}$$

For a constant image, its contrast value is equal to 0. In contrast, chaotic texture has a higher contrast value.

2. `Correlation` represents the correlation of pixels with various offsets and has an expression of

$$\texttt{Correlation} = \frac{1}{N_g} \sum_{[\Delta x, \Delta y]} \left(\sum_{i=1}^{L} \sum_{j=1}^{L} \frac{(i-\mu_i)(j-\mu_j)}{\sigma_i \sigma_j} \mathbf{G}_{[\Delta x, \Delta y]}[i,j] \right). \tag{4.3}$$

In Eq. (4.3), μ_i and μ_j represent GLCM means, and σ_i and σ_j are GLCM standard deviations. The definitions of GLCM means and standard deviations are listed as follows:

$$\begin{cases} \mu_i = \sum_{i=1}^{L} \sum_{j=1}^{L} i \cdot \mathbf{G}_{[\Delta x, \Delta y]}[i,j], \quad \sigma_i = \sqrt{\sum_{i=1}^{L} \sum_{j=1}^{L} (i-\mu_i)^2 \cdot \mathbf{G}_{[\Delta x, \Delta y]}[i,j]}, \\ \mu_j = \sum_{i=1}^{L} \sum_{j=1}^{L} j \cdot \mathbf{G}_{[\Delta x, \Delta y]}[i,j], \quad \sigma_j = \sqrt{\sum_{i=1}^{L} \sum_{i=1}^{L} (j-\mu_i)^2 \cdot \mathbf{G}_{[\Delta x, \Delta y]}[i,j]}. \end{cases} \tag{4.4}$$

The correlation value ranges from -1 to 1, which correspond to perfectly negative and positive correlation, respectively. For a constant image, its correlation value approaches infinity.

3. `Energy` sums squared entries in GLCMs as follows:

$$\texttt{Energy} = \frac{1}{N_g} \sum_{[\Delta x, \Delta y]} \left(\sum_{i=1}^{L} \sum_{j=1}^{L} \mathbf{G}^2_{[\Delta x, \Delta y]}[i,j] \right). \tag{4.5}$$

The constant image has the highest energy value of 1.

4. `Homogeneity` measures the closeness of the distribution of GLCM entries to the GLCM diagonal and is calculate as:

$$\texttt{Homogeneity} = \frac{1}{N_g} \sum_{[\Delta x, \Delta y]} \left(\sum_{i=1}^{L} \sum_{j=1}^{L} \frac{\mathbf{G}_{[\Delta x, \Delta y]}[i,j]}{1+|i-j|} \right). \tag{4.6}$$

The diagonal GLCM has the highest homogeneity value of 1.

4.1.2 Gradient of Texture (GoT)

To describe texture changes between objects, Wang et al. [103] proposed the GoT attribute. The GoT value of a given point measures perceptual dissimilarity between two square neighboring windows that share a side centered around the given point. For simplicity, in this chapter the point centered at the shared side is referred to as the center point. Figure 4.2 illustrates the changes of GoT values when the center point moves across two different texture regions separated by a purely vertical boundary. By sliding the center point and its two neighboring windows along the horizontal direction, we obtain the GoT profile following the curve shown at the bottom of the figure. The highest GoT value, which represents the highest dissimilarity, is achieved when the center point falls exactly on the texture boundary, where left window $\mathbf{W}_{x-}$ and right window $\mathbf{W}_{x+}$ have completely different texture content. When the center point is moved away from the texture boundary to the position shown in Figure 4.2, the content of $\mathbf{W}_{x+}$ is purely from the right texture region, while $\mathbf{W}_{x-}$ contains both textures. The texture partially shared by two windows reduces the dissimilarity and causes the decrease of the GoT value. If the center point keeps moving away from the boundary, we reach a point where the contents of $\mathbf{W}_{x-}$ and $\mathbf{W}_{x+}$ are from the same texture region. The exactly same texture in two windows leads to the zero value of the GoT.

Because of the purely vertical texture boundary in Figure 4.2, computing the GoT value

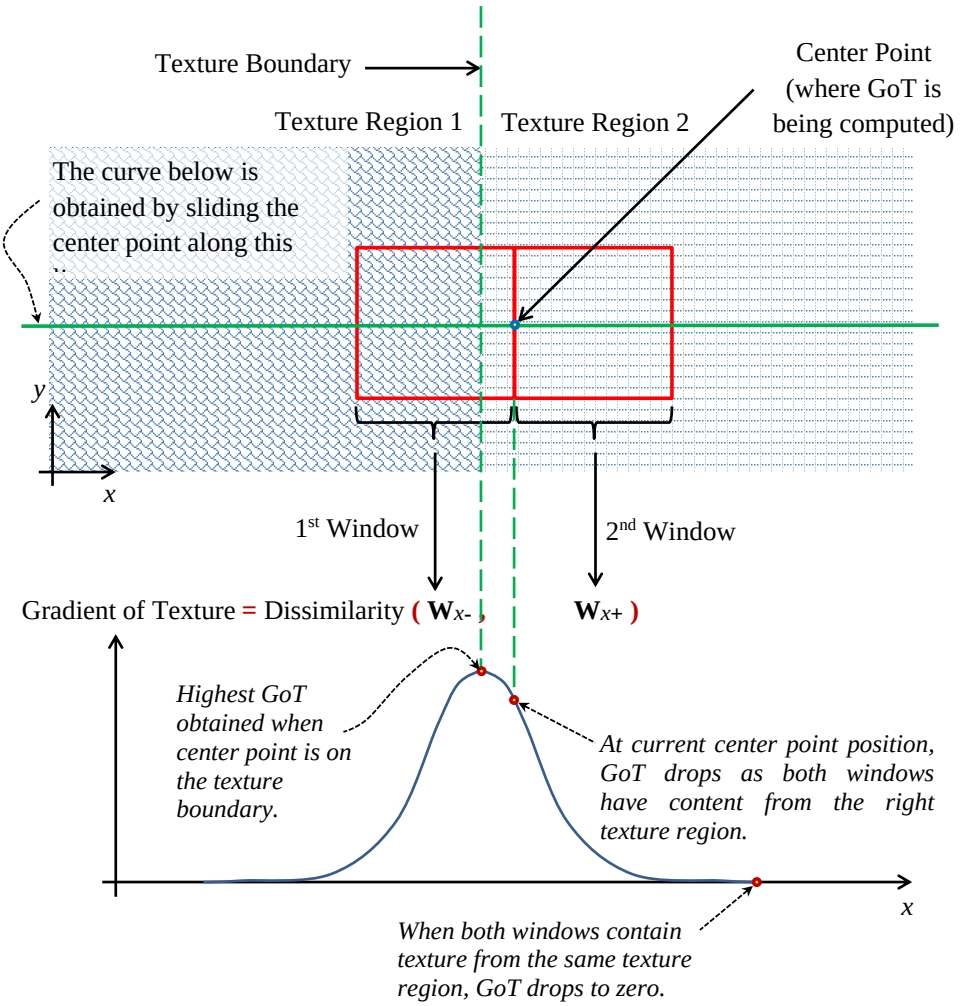


Figure 4.2: The GoT value at the center point represents texture dissimilarity between two square neighboring windows.

in the horizontal direction is sufficient. In general, the direction of the texture boundary is random, and the GoT value needs to involve components in both horizontal and vertical directions simultaneously. In addition, to capture scale variations in texture contents, neighboring windows with different sizes are used to calculate GoT values across different scales. Therefore, to improve robustness, we combine GoT values calculated along different directions and with different scales and obtain the multi-scale GoT shown as follows:

$$\begin{cases} \mathbf{G}_x[i,j] = \sum_{n=1}^{N} w_n \cdot d\left(\mathbf{W}_{n,x-}^{i,j}, \mathbf{W}_{n,x+}^{i,j}\right), \\ \mathbf{G}_y[i,j] = \sum_{n=1}^{N} w_n \cdot d\left(\mathbf{W}_{n,y-}^{i,j}, \mathbf{W}_{n,y+}^{i,j}\right), \\ \mathbf{G}[i,j] = \left(\mathbf{G}_x^2[i,j] + \mathbf{G}_y^2[i,j]\right)^{\frac{1}{2}}, \end{cases} \tag{4.7}$$

where $\mathbf{W}_{n,x-}^{i,j}$, $\mathbf{W}_{n,x+}^{i,j}$, $\mathbf{W}_{n,y-}^{i,j}$, and $\mathbf{W}_{n,y+}^{i,j}$ denote neighboring windows with a size of $(2n+1) \times (2n+1)$ located on the left, right, bottom, and top of point $[i,j]$, respectively, and $\mathbf{G}_x[i,j]$ and $\mathbf{G}_y[i,j]$ define GoT values at point $[i,j]$ along horizontal and vertical directions, respectively. In addition, $\mathbf{G}[i,j]$ represents the GoT value combining horizontal and vertical components, and function $d(\cdot)$ defines a dissimilarity measure. Finally, w_n, which denotes the weight function of the GoT value computed by $(2n+1) \times (2n+1)$ neighboring windows, is inversely proportional to n, e.g., $w_n = 1/n$ in [103].

4.2 Tensors and Multilinear Algebra

In the multilinear algebra, Nth-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ has N modes, and we denote its entries as $x_{i_1 i_2 \cdots i_N}$, where $i_1, i_2, \cdots, i_N$ represent indices in all modes. By unfolding tensor $\boldsymbol{\mathcal{X}}$ along the nth mode, we obtain matrix $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times (I_{n+1} I_{n+2} \cdots I_N I_1 \cdots I_{n-1})}$. Figure 4.3 illustrates the unfolding of a third-order tensor along three modes, where I_1, I_2, and I_3 represent the dimensions of three modes, respectively. The inverse operation of n-mode unfolding is n-mode folding, which restores tensor $\boldsymbol{\mathcal{X}}$ from matrix $\mathbf{X}_{(n)}$. The inner product of two tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ is defined as

$$\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_n} x_{i_1 i_2 \cdots i_n} \cdot y_{i_1 i_2 \cdots i_n}. \tag{4.8}$$

The inner product induces the Frobenius norm of a tensor, which is defined as $\|\boldsymbol{\mathcal{X}}\|_F = \sqrt{\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{X}} \rangle}$. The n-mode product of tensor $\boldsymbol{\mathcal{X}}$ by matrix $\mathbf{U} \in \mathbb{R}^{J_n \times I_n}$, denoted $\boldsymbol{\mathcal{X}} \times_n \mathbf{U}$,

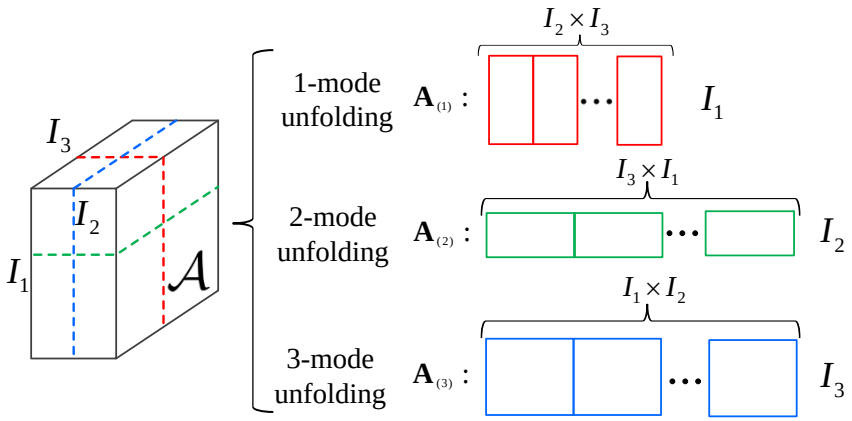


Figure 4.3: The unfolding of a third-order tensor along three modes.

defines new tensor $\mathcal{Y}$, the entries of which are calculated as

$$y_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n} x_{i_1 \cdots i_{n-1} i_n i_{n+1} \cdots i_N} \cdot u_{j_n i_n}. \tag{4.9}$$

An alternative way to implement the n-mode product is to fold $\mathbf{U} \cdot \mathbf{X}_{(n)}$ along the nth mode. The n-mode product satisfies the following properties:

1. Given tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and matrices $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{G} \in \mathbb{R}^{J_m \times I_m}$, we have

$$(\mathcal{A} \times_n \mathbf{F}) \times_m \mathbf{G} = (\mathcal{A} \times_m \mathbf{G}) \times_n \mathbf{F} = \mathcal{A} \times_n \mathbf{F} \times_m \mathbf{G}. \tag{4.10}$$

2. Given tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and matrices $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{G} \in \mathbb{R}^{K_n \times J_n}$, we have

$$(\mathcal{A} \times_n \mathbf{F}) \times_n \mathbf{G} = \mathcal{A} \times_n (\mathbf{G} \cdot \mathbf{F}). \tag{4.11}$$

To introduce the multilinear singular value decomposition (SVD) of Nth-order tensors, we start from the SVD of a matrix and rewrite its definition using notations in multilinear algebra as follows:

$$\mathbf{X} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^T = \mathbf{U}^{(1)} \cdot \mathbf{S} \cdot \mathbf{U}^{(2)^T} = \mathbf{S} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)}, \tag{4.12}$$

where $\mathbf{X}$ is a matrix with the dimension of $I_1 \times I_2$ and $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$, $n = 1, 2$, is a unitary matrix. $\mathbf{S}$ as a diagonal $I_1 \times I_2$ matrix can be written as $\mathbf{S} = \text{diag}\left(\sigma_1, \sigma_2, ..., \sigma_{\min(I_1, I_2)}\right)$, where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(I_1, I_2)} \geq 0$. Entry σ_i, $i = 1, 2, \cdots, \min(I_1, I_2)$, represent non-negative singular values of $\mathbf{X}$. As an extension of the SVD on matrices, the SVD of Nth-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, referred to as high-order SVD (HOSVD), is defined as

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{S}} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}, \tag{4.13}$$

where $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$, $n = 1, 2, \cdots, N$, is a unitary matrix, the column vectors of which span the column space of n-mode unfolding matrix $\mathbf{X}_{(n)}$ [118] and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the core tensor. We denote $\boldsymbol{\mathcal{S}}_{i_n = \alpha}$ as a subtensor of tensor $\boldsymbol{\mathcal{S}}$ with its nth order fixed as α, and tensor $\boldsymbol{\mathcal{S}}$ has the following properties:

1. Two subtensors $\boldsymbol{\mathcal{S}}_{i_n = \alpha}$ and $\boldsymbol{\mathcal{S}}_{i_n = \beta}$ are orthogonal for any n, α, and β subject to $\alpha \neq \beta$, the mathematical expression of which is shown as follows:

$$\langle \boldsymbol{\mathcal{S}}_{i_n = \alpha}, \boldsymbol{\mathcal{S}}_{i_n = \beta} \rangle = 0, \forall n \in \{1, 2, \cdots, N\}, \forall \alpha, \beta \in \{1, 2, \cdots, I_n\}, \text{and } \alpha \neq \beta \tag{4.14}$$

2. The Frobenius norms of nth order subtensors extracted from tensor $\boldsymbol{\mathcal{X}}$, denoted $\left\| \boldsymbol{\mathcal{S}}_{i_n = i} \right\|_F$, $i = 1, 2, \cdots, I_n$, are n-mode singular values of $\boldsymbol{\mathcal{X}}$, which are in an order of $\left\| \boldsymbol{\mathcal{S}}_{i_n = 1} \right\|_F \geq \left\| \boldsymbol{\mathcal{S}}_{i_n = 2} \right\|_F \geq \cdots \geq \left\| \boldsymbol{\mathcal{S}}_{i_n = I_n} \right\|_F \geq 0$.

In addition, the matrix representation of the HOSVD can be obtained as follows:

$$\mathbf{X}_{(n)} = \mathbf{U}^{(n)} \cdot \mathbf{S}_{(n)} \cdot \left(\mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n+2)} \otimes \cdots \otimes \mathbf{U}^{(N)} \otimes \mathbf{U}^{(1)} \otimes \mathbf{U}^{(2)} \otimes \cdots \otimes \mathbf{U}^{(n-1)} \right)^T, \tag{4.15}$$

where $\mathbf{S}_{(n)}$ is the n-mode unfolding matrix of $\boldsymbol{\mathcal{S}}$ and $\otimes$ denotes the Kronecker product [119].

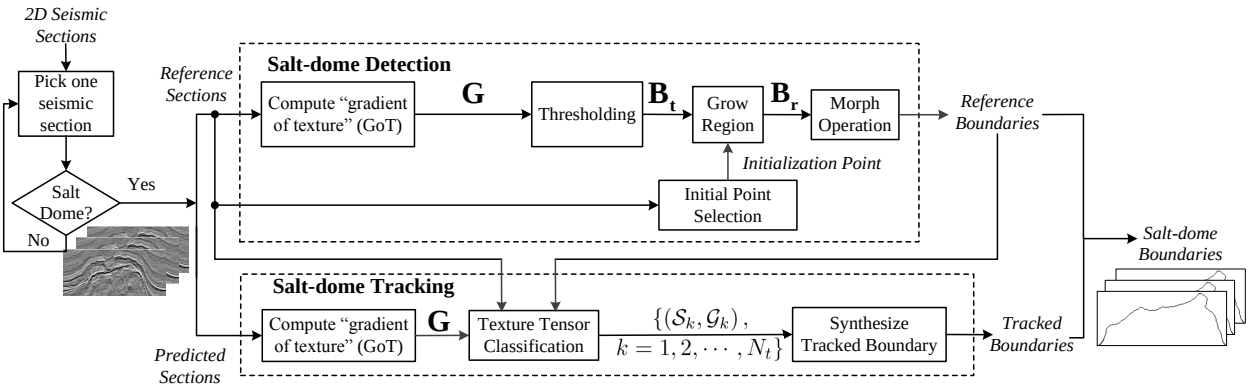


Figure 4.4: The diagram of the interactive salt-dome interpretation method.

4.3 Interactive Salt-dome Interpretation Framework

In this section, we focus on developing a framework that can label boundaries of salt domes both effectively and efficiently. Similar to interactive fault interpretation in the previous chapter, the salt-dome interpretation method also contains two main parts, salt-dome detection in reference sections and salt-dome tracking through predicted sections. We design a texture-attribute-based workflow, which is capable of detecting salt-dome boundaries with high accuracy and efficiency and robust to noise. Once salt-dome detection is performed in reference sections, we track initially detected boundaries through predicted sections to accomplish the labeling process. Without tweaking parameters for boundary detection in every seismic section, the tracking process is more efficient. The tracking algorithm automatically constructs tensors along initially detected boundaries using the noise-adjusted tensor-based incremental learning method. Features extracted from tensors capturing the strong correlation between similar texture patches are utilized to estimate tracked positions in predicted sections. An overall block diagram for the interactive salt-dome boundary labeling is depicted in Figure 4.4.

4.3.1 Salt-dome Detection in Reference Sections

The upper part of Figure 4.4 shows an overview of the salt-dome detection method. To capture the change in textures, we generate GoT map $\mathbf{G}$. Then, we apply a threshold on

$\mathbf{G}$ and obtain binary image $\mathbf{B}_t$, where the boundary regions of the salt dome are highlighted. From an initialization point inside the salt dome, we perform region growing, and regions indicating the salt dome keep growing and stop at boundary regions in $\mathbf{B}_t$. Finally, we perform binary morphological operations especially the closing operation to enhance grown regions $\mathbf{B}_r$, and enhanced regions determine the boundary of the salt dome. In the following sections, we will explain each module in detail.

Dissimilarity Measure in GoT

In the detection workflow, we calculate the multi-scale GoT maps [103] of seismic sections. In contrast to the salt body and other surrounding structures, the boundary regions of the salt dome indicating texture changes in seismic sections have higher GoT values. As we introduced in Sec. 4.1.2, the GoT value at a given point is determined by perceptual dissimilarity between neighboring windows. Since manual interpretation depends highly on interpreters' experience and expertise, to mimic such a subjective process, the design of the dissimilarity measure needs to involve the benefits of the human visual system (HVS). Therefore, when acquiring the GoT attribute, we choose the dissimilarity measure with an expression of

$$d\left(\mathbf{W}_{-}, \mathbf{W}_{+}\right) = E\left\{\left|\mathscr{F}\left\{\left|\mathscr{F}\left\{\left|\mathbf{W}_{-} - \mathbf{W}_{+}\right|\right\}\right|\right\}\right|\right\}, \tag{4.16}$$

where $\mathscr{F}\{\cdot\}$ denotes the 2D DFT, E is the expectation operator, and $\mathbf{W}_{-}$ and $\mathbf{W}_{+}$ denote two neighboring windows, respectively. This dissimilarity measure is a close variant of the one introduced in [120], which has been shown to be consistent with human perception while being computationally efficient. The dissimilarity measure characterizes the amount of variation or chaos in the magnitude spectrum of the absolute difference of two neighboring windows. The outer Fourier transform calculates the magnitude spectrum of the magnitude spectrum. Since such a dissimilarity measure is highly consistent with human perception [120], we believe it partially imitates interpreters' behaviors when delineating salt-dome boundaries.

Thresholding

On the obtained GoT map, we apply a threshold to highlight the boundary regions of the salt dome. The threshold value can be determined interactively with a human interpreter or automatically. To select a threshold value in an automatic way, we employ Otsu's method [121], which was designed based on the assumption that pixels in an image from two classes form a bimodal histogram. Therefore, Otsu's method determines the optimal threshold, denoted T_g, by minimizing the intra-class variance, which sums the weighted variances of two classes. The corresponding objective function is shown as follows:

$$T_g = \arg\min_{T} \left\{ \sigma_1^2(T) \sum_{i=0}^{T-1} p(i) + \sigma_2^2(T) \sum_{i=T}^{N-1} p(i) \right\}, \tag{4.17}$$

where T is the threshold value, $p(i)$ denotes the probability of grayscale value i in the histogram, and N represents the number of grayscale levels. In addition, $\sigma_1^2(T)$ and $\sigma_2^2(T)$ are the variances of first and second classes, respectively. The optimal threshold value is found by exhaustively searching between 0 and $N-1$. For more efficient implementation, Otsu [121] demonstrates that minimizing the intra-class variance is equivalent to maximizing the inter-class variance. Therefore, Eq. (4.17) can be simplified as

$$T_g = \arg\max_{T} \left\{ \left(\sum_{i=0}^{T-1} p(i) \right) \left(\sum_{i=0}^{T-1} p(i) \right) \left(\mu_1(T) - \mu_2(T) \right) \right\}, \tag{4.18}$$

where $\mu_1(T)$ and $\mu_2(T)$ are the means of first and second classes, respectively.

Initial Point Selection and Region Growing

After applying a threshod on the GoT map, we obtain binary image $\mathbf{B}_t$, where the boundary regions of the salt-dome can be highlighted. This process may inevitably involve other structures containing texture changes, although they are irrelevant to the salt dome. To ensure that the detection method focus only on salt-dome boundaries, we identify an initial-

ization point inside the salt dome and keep it growing until a highlighted boundary is met. The grown region is denoted as $\mathbf{B}_r$. The initialization point can be interactively selected with the help of interpreters. The interactive selection of the initialization point considerably reduces the time spent by interpreters since the interpreter needs only to quickly click on *any* arbitrary point within the salt region, in contrast to carefully traversing a long, tortuous salt-dome boundary. In addition, the interactive detection method is not sensitive to the position of the initialization point as long as it falls within the salt dome.

Morphological Operations

Dilation and erosion, as two basic morphological operations, can enlarge and shrink the regions of salt-dome boundaries highlighted in a binary image with structural element $\mathbf{H}$. The mathematical expressions of dilation and erosion are shown as follows:

$$\begin{cases} \text{Dilation:} \quad \mathbf{M} \oplus \mathbf{H} = \bigcup\limits_{z \in \mathbf{M}} \mathbf{H}_z \\ \text{Erosion:} \quad \mathbf{M} \ominus \mathbf{H} = \{z | \mathbf{H}_z \subseteq \mathbf{M}\} \end{cases}, \tag{4.19}$$

where $\mathbf{M}$ and $\mathbf{H}_z$ represent the binary image and the structural element centered at pixel z, respectively. The dilated region can be understood as the locus of the points covered by $\mathbf{H}$ when the center of $\mathbf{H}$ moves inside $\mathbf{M}$. In contrast, the eroded result represents the locus of points reached by the center of $\mathbf{H}$ when $\mathbf{H}$ moves inside $\mathbf{M}$. A dilation followed by an erosion using the same structuring element defines the closing operation, which is applied on $\mathbf{B}_r$ to obtain the smoothness and continuity of detected salt-dome boundaries in reference sections.

4.3.2 Salt-dome Tracking Through Predicted Sections

In the detection of salt-dome boundaries, the accuracy of labeled boundaries depends greatly on the selection of certain parameters. Since salt-dome structures commonly vary

across seismic sections, interpreters have to adjust parameters so that detected boundaries in each section can accurately capture structural changes. However, extra labor and time involved in the tuning of parameters may lower interpretation efficiency. Therefore, to avoid the frequent adjustment of parameters, we track detected boundaries in reference sections that constitute the minority of seismic sections through a seismic volume and synthesize salt-dome boundaries in predicted sections with limited human intervention. The lower part of Figure 4.4 illustrates the pipeline of the tracking method, each step of which is going to be introduced in detail in the following subsections.

Noise-adjusted Incremental PCA of 3-order Tensors

By observing salt domes in seismic sections as Figure 4.1 shows, we notice that local areas along salt-dome boundaries commonly have similar textures. Therefore, by classifying boundary textures in the reference section, we build texture tensors and employ learning algorithms to acquire their texture features, which will act as important constraints to the synthesis of salt-dome boundaries in predicted sections. Before we introduce the classification method on boundary textures, we introduce the noise-adjusted incremental PCA and its extension in the field of high-order tensors.

Since PCA is intimately related to the SVD, we can derive incremental PCA from the incremental SVD. The incremental SVD is an efficient and stable algorithm to update the SVD of an original matrix when new data arrives. We denote the original matrix with a dimension of $L \times M$ as $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_M]$, $\mathbf{a}_i \in \mathbb{R}^L$, $i = 1, 2, \cdots, M$, and the SVD factorizes $\mathbf{A}$ into $\mathbf{U_A}$, $\mathbf{S_A}$, and $\mathbf{V_A}$ as Eq. (4.12) shows. Similarly, we denote new data or observations as matrix $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_N]$, $\mathbf{b}_j \in \mathbb{R}^L$, $j = 1, 2, \cdots, N$, which has a dimension of $L \times N$. We concatenate the column vectors of matrices $\mathbf{A}$ and $\mathbf{B}$ and obtain updated matrix $\mathbf{X}$ in a form of $[\mathbf{A}|\mathbf{B}] = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_M, \mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_N]$, which has a dimension of $L \times (M + N)$. The incremental SVD algorithm [122] updates the SVD of $\mathbf{X}$, denoted $\mathbf{U_X} \cdot \mathbf{S_X} \cdot \mathbf{V_X}$, with high efficiency on the basis of $\{\mathbf{U_A}, \mathbf{S_A}, \mathbf{V_A}\}$ and $\mathbf{B}$. The

details of the incremental updating process are listed as follows:

1. The column mean vector of matrix $\mathbf{X}$, denoted $\mu_{\mathbf{X}}$, is updated as

$$\mu_{\mathbf{X}} = \frac{M}{M+N}\mu_{\mathbf{A}} + \frac{N}{M+N}\mu_{\mathbf{B}}, \tag{4.20}$$

   where $\mu_{\mathbf{A}}$ and $\mu_{\mathbf{B}}$ are column mean vectors of matrices $\mathbf{A}$ and $\mathbf{B}$, respectively;

2. New matrix $\widehat{\mathbf{B}}$ is derived from $\mathbf{B}$ by removing $\mu_{\mathbf{B}}$ from new observations and appending the difference of $\mu_{\mathbf{A}}$ and $\mu_{\mathbf{B}}$ as follows:

$$\widehat{\mathbf{B}} = \left[(\mathbf{B} - \mu_{\mathbf{B}} \cdot \mathbf{1}_{1\times N}) \left| \sqrt{\frac{MN}{M+N}} (\mu_{\mathbf{A}} - \mu_{\mathbf{B}}) \right. \right], \tag{4.21}$$

   where $\mathbf{1}_{1\times N}$ is a matrix of ones with the size of $1 \times N$;

3. Matrix $\left(\widehat{\mathbf{B}} - \mathbf{U_A}\mathbf{U_A}^T\widehat{\mathbf{B}}\right)$ represents the components of $\widehat{\mathbf{B}}$ not already in subspace $\mathbf{U_A}$. The orthonormal basis that span the column space of matrix $\left(\widehat{\mathbf{B}} - \mathbf{U_A}\mathbf{U_A}^T\widehat{\mathbf{B}}\right)$ constructs matrix $\widetilde{\mathbf{B}}$, which can be obtained by applying the QR decomposition on matrix $\left(\widehat{\mathbf{B}} - \mathbf{U_A}\mathbf{U_A}^T\widehat{\mathbf{B}}\right)$.

4. Matrix $\mathbf{R}$ constructed by $\widetilde{\mathbf{B}}$, $\widehat{\mathbf{B}}$, $\mathbf{U_A}$, and $\mathbf{S_A}$ has a form of

$$\mathbf{R} = \begin{bmatrix} \mathbf{S_A} & \mathbf{U_A}^T\widehat{\mathbf{B}} \\ \mathbf{0} & \widetilde{\mathbf{B}}\left(\widehat{\mathbf{B}} - \mathbf{U_A}\mathbf{U_A}^T\widehat{\mathbf{B}}\right) \end{bmatrix}, \tag{4.22}$$

   and its SVD is denoted as $\mathbf{U_R} \cdot \mathbf{S_R} \cdot \mathbf{V_R}^T$;

5. The SVD of $\mathbf{X}$ is updated by $\mathbf{U_X} = \left[\mathbf{U_A}|\widetilde{\mathbf{B}}\right]\mathbf{U_R}$, $\mathbf{S_X} = \mathbf{S_R}$, and $\mathbf{V_X} = \begin{bmatrix} \mathbf{V_A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}\mathbf{V_R}$.

Therefore, by keeping the first K column vectors in $\mathbf{U_X}$, which correspond to the largest K singular values in $\mathbf{S_X}$, we obtain projection matrix $\widetilde{\mathbf{U}}_{\mathbf{X}} = [\mathbf{u}_1, \mathbf{u_2}, \cdots, \mathbf{u}_K]$ for incremental PCA.

In the salt-dome tracking method, we build 3-order tensors along detected boundaries in reference sections. To acquire texture features from tensors with high efficiency, we extend the the incremental SVD of matrices to higher orders and implement the tensor-based incremental SVD by unfolding tensors in each mode. Figure 4.5 illustrates the unfolding matrices of a 3-order tensor containing original and newly arrived data. To distinguish with original data, in Figure 4.5 we mark newly arrived data as meshed volumes or blocks. Given that tensor $\mathcal{A}$ is built by stacking image patches along the third mode, when new data arrive, we append them to original data in the same way. However, in unfolding matrices along different modes the positions of new data are different. Therefore, we have a specific strategy for each unfolding matrix to ensure that we can update its SVD result incrementally. Unfolding matrix $\mathbf{A}_{(1)}$ simply concatenates the matrices of original and new data, and its SVD result can be directly updated by the incremental SVD method introduced above. In contrast, unfolding matrix $\mathbf{A}_{(2)}$ has a form of alternating original and new data as Figure 4.5 shows. Using a permutation matrix, we append column vectors from new data to those of $\mathbf{A}_{(2)}$ without changing the column space. Rearranged $\mathbf{A}_{(2)}$ has the same structure with $\mathbf{A}_{(1)}$, and its SVD result can be updated by the incremental SVD method as well. However, as Figure 4.5 depicts, unfolding matrix $\mathbf{A}_{(3)}$ fails to follow the layouts of $\mathbf{A}_{(1)}$ and rearranged $\mathbf{A}_{(2)}$. Instead of appending column vectors, the appearance of new data increases the number of row vectors in $\mathbf{A}_{(3)}$, which leads to the change of the column space. The SVD, which extracts the orthonormal basis of the column space, cannot be incrementally updated in $\mathbf{A}_{(3)}$. To solve this problem, we incrementally update the SVD of $\mathbf{A}_{(3)}^T$ rather than $\mathbf{A}_{(3)}$. Since tensor $\mathcal{A}$ is constructed by stacking image patches along the third mode, each row vector in $\mathbf{A}_{(3)}$ actually represents a vectorized image patch. Features extracted from the row space of $\mathbf{A}_{(3)}$ describe the patterns of entire image patches, which are more representative than those extracted from the column space of $\mathbf{A}_{(3)}$. Therefore, when new data arrive, we incrementally update the SVDs of $\mathbf{A}_{(1)}$, rearranged $\mathbf{A}_{(2)}$, and $\mathbf{A}_{(3)}^T$ and derive projection matrices for the PCA of each unfolding matrix, denoted

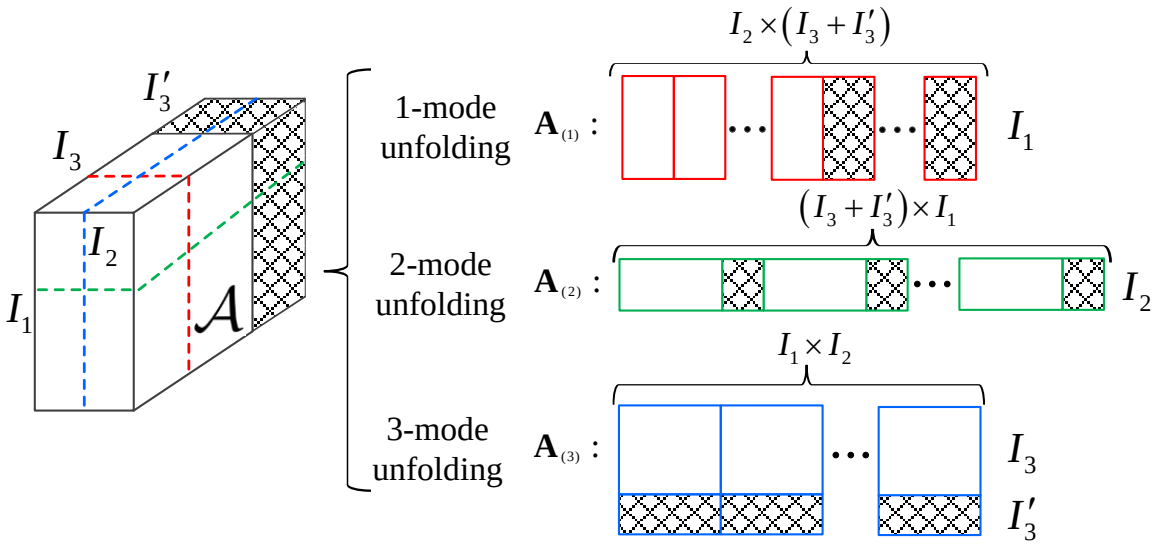


Figure 4.5: Unfolding matrices of a 3-order tensor containing original and newly arrived data. In unfolding matrices along different modes, the positions of new data are different.

$\widetilde{\mathbf{U}}^{(1)} \in \mathbb{R}^{I_1 \times P_1}$, $\widetilde{\mathbf{U}}^{(2)} \in \mathbb{R}^{I_2 \times P_2}$, and $\widetilde{\mathbf{U}}^{(3)} \in \mathbb{R}^{I_1 I_2 \times P_3}$, respectively. $P_1 < I_1$, $P_2 < I_2$, and $P_3 < I_1 I_2$ correspond to the numbers of the largest singular values selected from unfolding matrices, respectively.

Seismic interpretation is commonly conducted on migrated data, which are generated by passing raw seismic data through a series of complicated processing steps. In seismic data acquisition and processing systems, seismic data easily involve various types and levels of noise. Low signal-to-noise ratios (SNRs) may influence features extracted from tensors built from detected boundaries in reference sections and further reduce the accuracy of tracked boundaries. To deal with this problem, on the basis of Roger's work [123], we introduce noise-adjusted tensor-based incremental PCA that ranks principal components (PCs) along different modes by the SNR rather than the variation. Since noise is commonly assumed to be additive, we define image patches extracted from seismic sections as $\mathbf{X} = \mathbf{A} + \mathbf{N}$, where $\mathbf{A}$ is the image patch without noise and $\mathbf{N}$ represents noise that are independent to $\mathbf{A}$. We denote the covariance matrices of $\mathbf{X}$, $\mathbf{A}$, and $\mathbf{N}$ as $\mathbf{\Sigma_X}$, $\mathbf{\Sigma_A}$, and $\mathbf{\Sigma_N}$, respectively. In this chapter, we estimate $\mathbf{\Sigma_N}$ using the correlation method proposed in [124]. To arrange PCs in the order of SNRs, we need to apply a noise-whitening process

that transforms $\mathbf{N}$ to identity covariance matrix $\widehat{\mathbf{N}}$. Therefore, we define transform matrix $\mathbf{P}$ and analyze the noise-whitening process as follows:

$$\mathbf{\Sigma}_{\widehat{\mathbf{N}}} = \mathbf{P}^T \mathbf{\Sigma}_{\mathbf{N}} \mathbf{P} = \mathbf{P}^T \mathbf{V} \mathbf{\Delta}_{\mathbf{N}} \mathbf{V}^T \mathbf{P} = \mathbf{I}, \tag{4.23}$$

where orthonormal matrix $\mathbf{V}$ contains the eigenvectors of $\mathbf{\Sigma}_{\mathbf{N}}$. To satisfy the constraint in Eq. (4.23), we yield transform matrix $\mathbf{P} = \mathbf{V}\mathbf{\Delta}_{\mathbf{N}}^{-1/2}$. In the tracking method, we will use the noise-whitening process to remove the influence of different noise levels in the reference and predicted sections and improve the accuracy of tracked salt-dome boundaries.

Adaptive Texture Tensor Classification

To extract texture features from salt-dome boundaries detected in a reference section, we define pairs of patches centered at boundary points that are subimages containing the boundary regions of the reference section and its corresponding multi-scale GoT map. Since matrices are particular third-order tensors with a third-mode dimension equal to one, we denote patch pairs in a form of tensor pairs as $\{\boldsymbol{\mathcal{S}}_{p_i}, \boldsymbol{\mathcal{G}}_{p_i}\} \subset \mathbb{R}^{I_1 \times I_2 \times 1}$, $i = 1, 2, \cdots, N_b$, where N_b represents the number of all boundary points in the reference section and I_1 and I_2 define the dimensions of patches along the depth and crossline directions, respectively. Since textures have no bias on either the depth or crossline direction, we commonly assume that I_1 is equal to I_2. In addition, to ensure that boundary textures can be captured by square-shaped patches, we set the edge length of patches as one-tenth of the larger of the vertical and horizontal resolutions. Furthermore, by evaluating the similarity between patches using tensor-based incremental learning, we can classify boundary textures and build a set of texture tensors, denoted $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$, $k = 1, 2, \cdots, N_t$.

To clarify the iterative classification strategy, we depict its block diagram in Figure 4.6. Because of the roughly semi-circular shape of detected salt-dome boundaries in the reference section, we traverse clockwise along boundary points and define p_1 as the first point

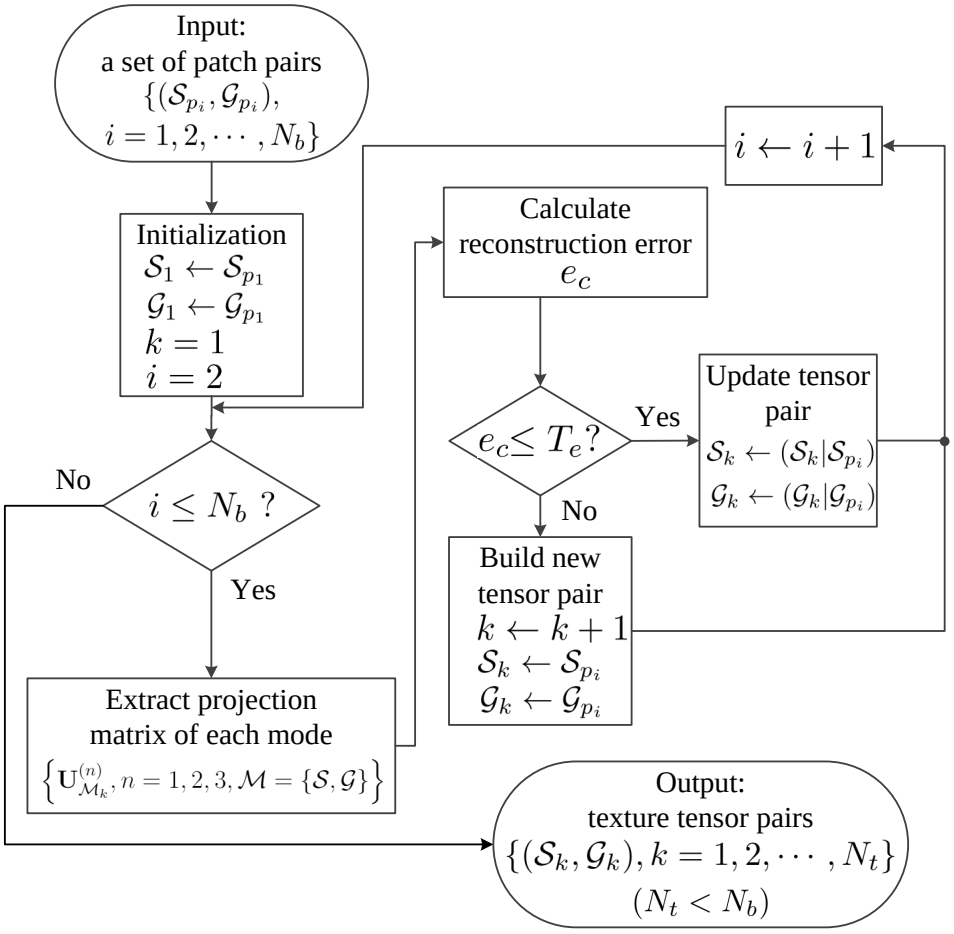
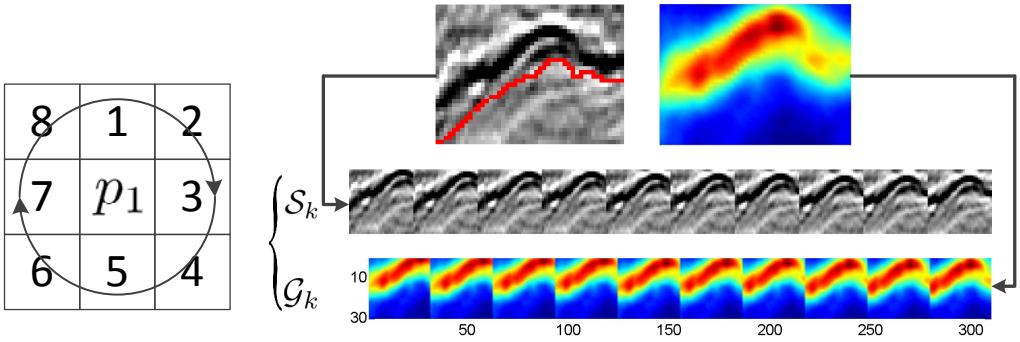


Figure 4.6: The block diagram of the adaptive classification of texture tensors.

in the bottom-left corner satisfying the dimension constraint of patches. After initializing $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{G}}_1\}$ with patch pair $\{\boldsymbol{\mathcal{S}}_{p_1}, \boldsymbol{\mathcal{G}}_{p_1}\}$, we search for p_2 in the 3×3 neighborhood of p_1 on the basis of the point priority shown in Figure 4.7(a), where a smaller number represents a higher priority. To determine whether the patch pair of p_2, $\{\boldsymbol{\mathcal{S}}_{p_2}, \boldsymbol{\mathcal{G}}_{p_2}\}$, are similar to patches in the initial tensor pair, $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{G}}_1\}$, we apply the tensor-based incremetal PCA introduced above to $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{G}}_1\}$ and obtain the projection matrix of each mode, denoted $\mathbf{U}^{(n)}_{\boldsymbol{\mathcal{M}}_1} \in \mathbb{R}^{I_n \times P_n}$, $(n = 1, 2)$, $\mathbf{U}^{(3)}_{\boldsymbol{\mathcal{M}}_1} \in \mathbb{R}^{(I_1 \times I_2) \times P_3}$, $\boldsymbol{\mathcal{M}} = \{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{G}}\}$, respectively, in which $[P_1, P_2, P_3]$ are determined empirically. Since projection matrices can extract texture features from tensors, we define reconstruction error e_c on the basis of projection matrices to to evaluate the sim-



(a) (b)

Figure 4.7: (a) The priority of points in the 3×3 neighborhood of p_1 and (b) The 1-mode unfolding matrices of a tensor pair extracted from the local area of a seismic section and its corresponding GoT map.

ilarity between patches and tensors as follows:

$$e_c = \sum_{n=\{1,2\},\boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \left\| \boldsymbol{\mathcal{M}}_{p_i} - \boldsymbol{\mathcal{M}}_{p_i} \times_n \left(\mathbf{U}^{(n)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathbf{U}^{(n)^T}_{\boldsymbol{\mathcal{M}}_k} \right) \right\|_F^2 + \sum_{\boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \left\| \mathbf{M}^{(3)}_{p_i} - \mathbf{M}^{(3)}_{p_i} \cdot \mathbf{U}^{(3)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathbf{U}^{(3)^T}_{\boldsymbol{\mathcal{M}}_k} \right\|_F^2 , \tag{4.24}$$

where p_i and k represent the indices of the current boundary point and tensor pair, respectively. If e_c is less than threshold T_e, the strong similarity allows us to extend $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{G}}_1\}$ by stacking $\{\boldsymbol{\mathcal{S}}_{p_2}, \boldsymbol{\mathcal{G}}_{p_2}\}$ along the third mode. Otherwise, the great deviation requires us to keep $\{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{G}}_1\}$ unchanged and initialize another tensor pair $\{\boldsymbol{\mathcal{S}}_2, \boldsymbol{\mathcal{G}}_2\}$ with $\{\boldsymbol{\mathcal{S}}_{p_2}, \boldsymbol{\mathcal{G}}_{p_2}\}$. By repeating the step above, we obtain classified tensor pairs that contain the local texture information of the detected boundary in the reference section. Figure 4.7(b) illustrates one local area of a seismic section and its corresponding GoT map, in which the red curve represents the labeled salt-dome boundary. In addition, Figure 4.7(b) shows the 1-mode unfolding matrices of a tensor pair extracted from this local area, in which texture patches have strong correlations. The entire algorithm of the adaptive texture-tensor classification is shown in Algorithm 2, where function $(\cdot|\cdot)$ describes an operation that concatenates tensors along the third mode.

Algorithm 2 Adaptive Classification of Texture Tensors

Input: a set of patch pairs $\{(\boldsymbol{\mathcal{S}}_{p_i}, \boldsymbol{\mathcal{G}}_{p_i}), i = 1, 2, \cdots, N_b\}$
Output: texture tensor pairs $\{(\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k), k = 1, 2, \cdots, N_t\}, N_t < N_b$

Initialization: $\boldsymbol{\mathcal{S}}_1 \leftarrow \boldsymbol{\mathcal{S}}_{p_1}$, $\boldsymbol{\mathcal{G}}_1 \leftarrow \boldsymbol{\mathcal{G}}_{p_1}$, and $k = 1$
for $i \leftarrow 2$ to N_b **do**
  • Calculate $\left\{ \mathrm{U}^{(n)}_{\boldsymbol{\mathcal{M}}_k},\ n = 1, 2, 3,\ \boldsymbol{\mathcal{M}} = \{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{G}}\} \right\}$ from $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$
  • Calculate reconstruction errors:
  $$e = \sum_{n=\{1,2\}, \boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \left\| \boldsymbol{\mathcal{M}}_{p_i} - \boldsymbol{\mathcal{M}}_{p_i} \times_n \left(\mathrm{U}^{(n)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathrm{U}^{(n)^T}_{\boldsymbol{\mathcal{M}}_k} \right) \right\|_F^2 +$$
  $$\sum_{\boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \left\| \mathrm{M}^{(3)}_{p_i} - \mathrm{M}^{(3)}_{p_i} \cdot \mathrm{U}^{(3)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathrm{U}^{(3)^T}_{\boldsymbol{\mathcal{M}}_k} \right\|_F^2$$
  if $e \leq T_e$ **then**
    $\boldsymbol{\mathcal{S}}_k \leftarrow (\boldsymbol{\mathcal{S}}_k | \boldsymbol{\mathcal{S}}_{p_i})$, and $\boldsymbol{\mathcal{G}}_k \leftarrow (\boldsymbol{\mathcal{G}}_k | \boldsymbol{\mathcal{G}}_{p_i})$, and $i \leftarrow i + 1$
  else
    $k \leftarrow k + 1$
    $\boldsymbol{\mathcal{S}}_k \leftarrow \boldsymbol{\mathcal{S}}_{p_i}$, $\boldsymbol{\mathcal{G}}_k \leftarrow \boldsymbol{\mathcal{G}}_{p_i}$
  end if
end for

Tracked Boundary Synthesis

On the basis of texture features extracted from classified texture tensors, we can localize boundary points in predicted sections. We first project the detected boundary in the reference section onto the target section and keep the coordinates of all points unchanged. Then, to identify the tracked point of every projected point p, we search among candidate points located along the normal direction of the projected boundary within radius R_s, which is determined by the inline number difference between the reference and current predicted sections. To define the textures of candidate points, we extract pairs of patches centered at these points from the predicted section and its corresponding GoT map, denoted $\{\boldsymbol{\mathcal{S}}_{p,j}, \boldsymbol{\mathcal{G}}_{p,j}\} \subset \mathbb{R}^{I_1 \times I_2 \times 1}$, $j = 1, 2, \cdots, (2R_s+1)$, where j represents the index of the candidate point. Since in the reference section the patch pair of projected point p belongs to one tensor pair $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$, $k \in \{1, 2, \cdots, N_t\}$, by comparing the similarity between $\{\boldsymbol{\mathcal{S}}_{p,j}, \boldsymbol{\mathcal{G}}_{p,j}\}$ and $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$, we can determine tracked point p^*. The block diagram of identifying tracked points is shown in Figure 4.8. Reconstruction error e_t is calculated as follows:

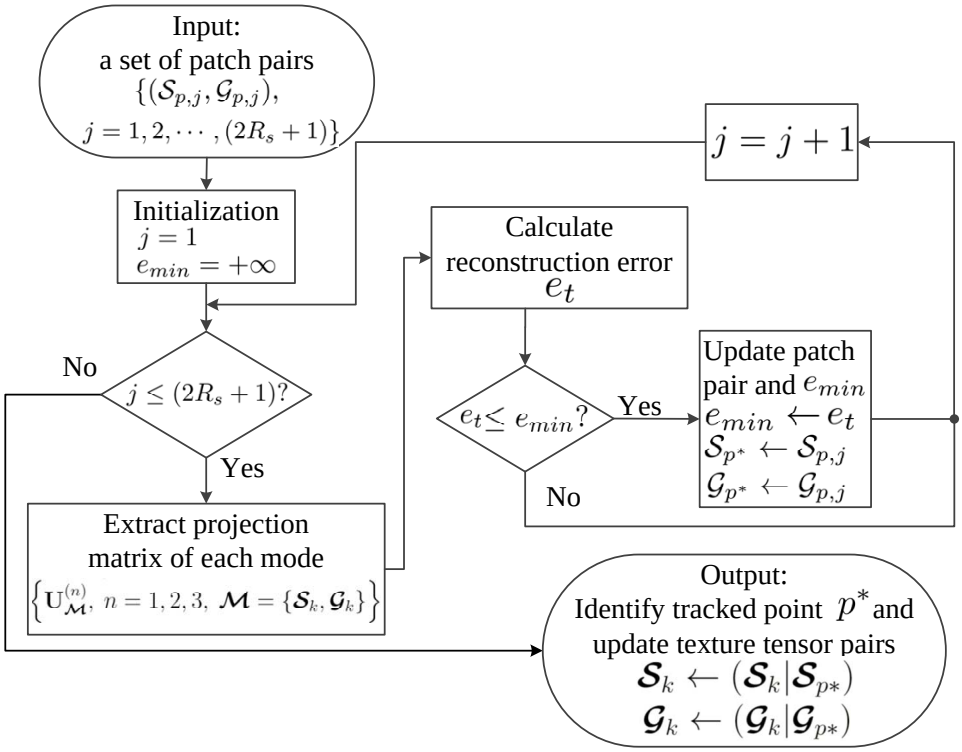


Figure 4.8: The block diagram of the identification of tracked points.

$$e_t = \sum_{n=\{1,2\}, \mathcal{M}=\{\mathcal{S},\mathcal{G}\}} \lambda_{\mathcal{M}} \cdot \left\| \mathcal{M}_{p,j} - \mathcal{M}_{p,j} \times_n \left(\mathrm{U}^{(n)}_{\mathcal{M}_k} \cdot \widetilde{\mathrm{U}}^{(n)^T}_{\mathcal{M}_k} \right) \right\|^2_F + \sum_{\mathcal{M}=\{\mathcal{S},\mathcal{G}\}} \lambda_{\mathcal{M}} \cdot \left\| \mathrm{M}^{(3)}_{p,j} - \mathrm{M}^{(3)}_{p,j} \cdot \mathrm{U}^{(3)}_{\mathcal{M}_k} \cdot \mathrm{U}^{(3)^T}_{\mathcal{M}_k} \right\|^2_F , \tag{4.25}$$

where $\lambda_{\mathcal{S}} = 1$ and $\lambda_{\mathcal{G}} = |\log(\mathbf{G}[i,j])|$ represent the weights of texture features extracted from seismic sections and corresponding GoT maps, respectively. $[i,j]$ denote the coordinates of the candidate point. Since GoT maps have been normalized to a range of zero to one, a larger $\mathbf{G}[i,j]$ indicates a smaller weight, which can move tracked points towards the real salt-dome boundaries of predicted sections by lowering the reconstruction error. Finally, the candidate point with the smallest reconstruction error is selected as the tracked point. The algorithm of localizing tracked points is shown in Algorithm 3. We apply the same process to all boundary points and obtain binary image $\mathbf{B}$, which contains initial tracked points.

Because of the complicated structures of salt domes, it is inevitable that tracked points

Algorithm 3 The Localization of Tracked Points

Input: a set of patch pairs $\{(\boldsymbol{\mathcal{S}}_{p,j}, \boldsymbol{\mathcal{G}}_{p,j}), j = 1, 2, \cdots, (2R_s + 1)\}$
Output: tracked point p^* and updated tensors $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$, $k \in \{1, 2, \cdots, N_t\}$

Initialization: $e_{min} = +\infty$
for $j \leftarrow 1$ to $(2R_s + 1)$ **do**
  • Calculate $\left\{\mathbf{U}^{(n)}_{\boldsymbol{\mathcal{M}}},\ n = 1, 2, 3,\ \boldsymbol{\mathcal{M}} = \{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}\right\}$ from $\{\boldsymbol{\mathcal{S}}_k, \boldsymbol{\mathcal{G}}_k\}$
  • Calculate reconstruction errors:
$$e = \sum_{n=\{1,2\}, \boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \lambda_{\boldsymbol{\mathcal{M}}} \cdot \left\| \boldsymbol{\mathcal{M}}_{p,j} - \boldsymbol{\mathcal{M}}_{p,j} \times_n \left(\mathbf{U}^{(n)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathbf{U}^{(n)^T}_{\boldsymbol{\mathcal{M}}_k} \right) \right\|^2_F +$$
$$\sum_{\boldsymbol{\mathcal{M}}=\{\boldsymbol{\mathcal{S}},\boldsymbol{\mathcal{G}}\}} \lambda_{\boldsymbol{\mathcal{M}}} \cdot \left\| \mathbf{M}^{(3)}_{p,j} - \mathbf{M}^{(3)}_{p,j} \cdot \mathbf{U}^{(3)}_{\boldsymbol{\mathcal{M}}_k} \cdot \mathbf{U}^{(3)^T}_{\boldsymbol{\mathcal{M}}_k} \right\|^2_F$$
  if $e \leq e_{min}$? **then**
    $e_{min} \leftarrow e$, $\boldsymbol{\mathcal{S}}_{p^*} \leftarrow \boldsymbol{\mathcal{S}}_{p,j}$, and $\boldsymbol{\mathcal{G}}_{p^*} \leftarrow \boldsymbol{\mathcal{G}}_{p,j}$
  end if
end for
$\boldsymbol{\mathcal{S}}_k \leftarrow (\boldsymbol{\mathcal{S}}_k | \boldsymbol{\mathcal{S}}_{p*})$, and $\boldsymbol{\mathcal{G}}_k \leftarrow (\boldsymbol{\mathcal{G}}_k | \boldsymbol{\mathcal{G}}_{p*})$

involve some outliers that may harm the smoothness and continuity of tracked boundaries. To remove these outliers, we apply the median filter to $\mathbf{B}$ and obtain filtered image $\widehat{\mathbf{B}}$ as follows:

$$\widehat{\mathbf{B}}[i,j] = \begin{cases} 0 & \mathbf{B}[i,j] - \underset{R_m \times R_m}{\text{median}}(\mathbf{B}[i,j]) = -1 \\ \underset{R_m \times R_m}{\text{median}}(\mathbf{B}[i,j]) & \text{Otherwise} \end{cases}, \tag{4.26}$$

where function $\underset{R_m \times R_m}{\text{median}}(\cdot)$ applies the median filter to the $R_m \times R_m$ analysis window of point $[i, j]$. Without involving new points generated by median filtering, Eq. (4.26) removes outliers and ensures the accuracy of the remaining tracked points. Furthermore, we connect the remaining tracked points clockwise with straight lines and obtain the initial labeling of the salt dome boundary. Since the connection based on lines may result in a jagged shape of the labeled boundary, which is geologically unreasonable, to delineate the salt-dome boundary more accurately, we employ two post-processing steps, dilation and skeletonization. We dilate initially labeled result $\mathbf{L}$ using a disk-shaped structural element. By applying skeletonization [125] to the dilated region, we can extract the tracked bound-

ary.

4.3.3 Similarity Index Measurement of Salt-dome Boundaries

By employing the salt-dome detection and tracking method introduced above, we can delineate salt-dome boundaries in seismic sections semi-automatically. To evaluate the performance of the interpretation method, we introduce a salt-dome similarity (SalSIM) index that measures the distances between semi-automatically labeled results and the ground truth provided by experienced interpreters. To explain the definition of the SalSIM index, in Figure 4.9 we draw two types of curves for illustration. The SalSIM index is developed based on the Fréchet distance [114], which can more accurately measure the deviation between two curves than the Haudorff distance [115] by taking the continuity of curves into account. On the basis of the Fréchet distance, we attempt to numerically describe local and global deviations between labeled results and the ground truth using local and global items in the SalSIM index, respectively. To compare the local details of labeled boundaries and the ground truth, we define an analysis window that identifies a pair of curve segments, as the rectangular shown in Figure 4.9. Therefore, by moving the analysis window along the ground truth and calculating the Fréchet distance of every pair of local segments, we can obtain a sequence of distances, $\mathbf{d} = [d_i]$, $i = 1, 2, \cdots, N_d$, where N_d is the total number of pairs. The mean and standard deviation of $\mathbf{d}$, denoted μ_d and σ_d, respectively, constitute the local item of the SalSIM index. In addition, the global item involves the Fréchet distance

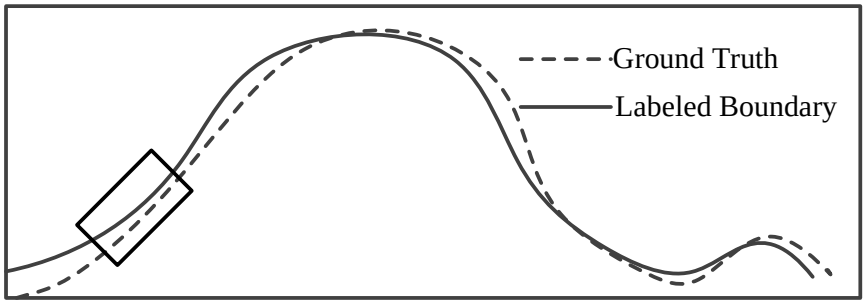


Figure 4.9: Local comparisons between the salt-dome boundary labeled by the interactive interpretation method and the ground truth labeled manually

of two entire curves, denoted d_{max}, which equals the largest distance in $\mathbf{d}$. Therefore, on the basis of obtained statistics, we introduce the SalSIM index as

$$\text{SalSIM} = \underbrace{e^{-\alpha\cdot(\mu_d+\sigma_d)}}_{\text{Local Item}} \cdot \underbrace{e^{-\beta\cdot d_{max}}}_{\text{Global Item}}, \tag{4.27}$$

where α and β are normalization factors determined empirically. Since the exponential function defined on negative real numbers has a range of zero to one, we apply it in the SalSIM index for normalization. Therefore, according to the expression of SalSIM in Eq. (4.27), a greater SalSIM value represents smaller deviation between the semi-automatically labeled salt-dome boundary and the ground truth and vice versa.

4.4 Experimental Results

In this section, we evaluate the performance of our detection and tracking method on a real seismic volume acquired from the Netherlands offshore F3 block with the size of 24×16 km^2 in the North Sea [116]. We focus on a local volume containing a distinguishable salt dome that has the inline number ranging from #389 to #409, the crossline number ranging from #401 to #701, and the time interval between 1,300ms and 1,848ms sampled every 4ms. Figure 4.10 illustrates a seismic section, Inline #400, extracted from the local volume.

4.4.1 Salt-dome Detection

The salt-dome detection method delineates the boundaries of salt domes based on the GoT maps of seismic sections, in which the point value represents the dissimilarity of neighboring square windows. To capture the multi-scale texture contrast of neighborhoods, we calculate GoT values by varying the window size from 3×3 to 11×11 and averaging the corresponding dissimilarity measures, denoted $\mathbf{G}$ in Eq. (4.7). Figure 4.11(a) shows the GoT map of Inline #400, in which likely boundary regions have higher GoT values. On

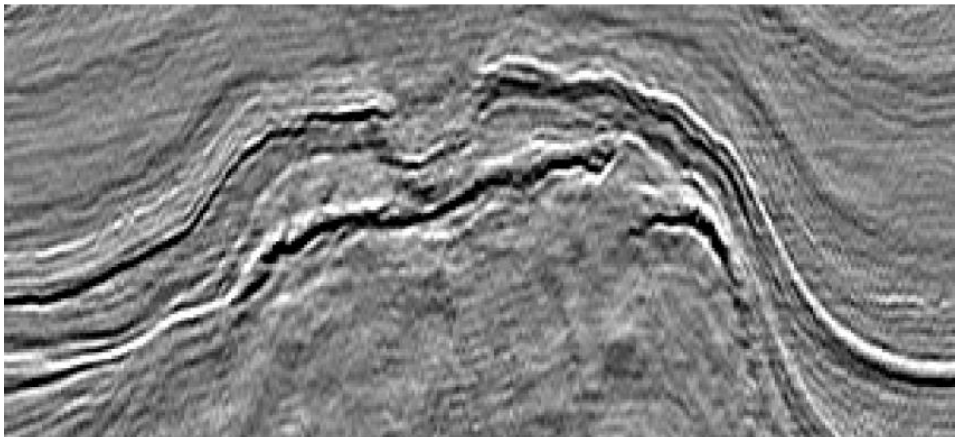


Figure 4.10: A seismic section (Inline #400) of the local volume contains the cross-section of a salt dome.

the basis of GoT maps, we utilize the detection workflow based on region growing to identify salt-dome areas and their corresponding boundaries. In the growing process, we tweak threshold T_g for each seismic section to yield the best growing result. In Figure 4.11(b), we compare the green detected boundary with the manually labeled red ground truth and notice that the former overlaps the most part of the latter except the bottom-left and -right corners.

Subjective Comparison of Detected Boundaries

Since GoT maps mainly characterize the texture variations of local neighborhoods in seismic sections, to demonstrate the benefit of the GoT attribute, we compare it with two conventional seismic attributes, the GLCM contrast attribute [22] and the gradient attribute [126], both of which have been introduced to detect salt-dome boundaries [30, 99]. To ensure fair comparison, we apply the same detection workflow to extract salt-dome boundaries from different attribute maps. In the detection workflow based on the GLCM contrast, for each point $[i, j]$, within its $(2R_d + 1) \times (2R_d + 1)$ analysis window, we can obtain a series of GLCMs by varying the directions and pixel distances of the offset. In the tested local volume, we obtain the GLCM contrast maps of seismic sections by setting $R_d = 4$. Figure 4.11(c) illustrates the GLCM contrast attribute of Inline #400, in which

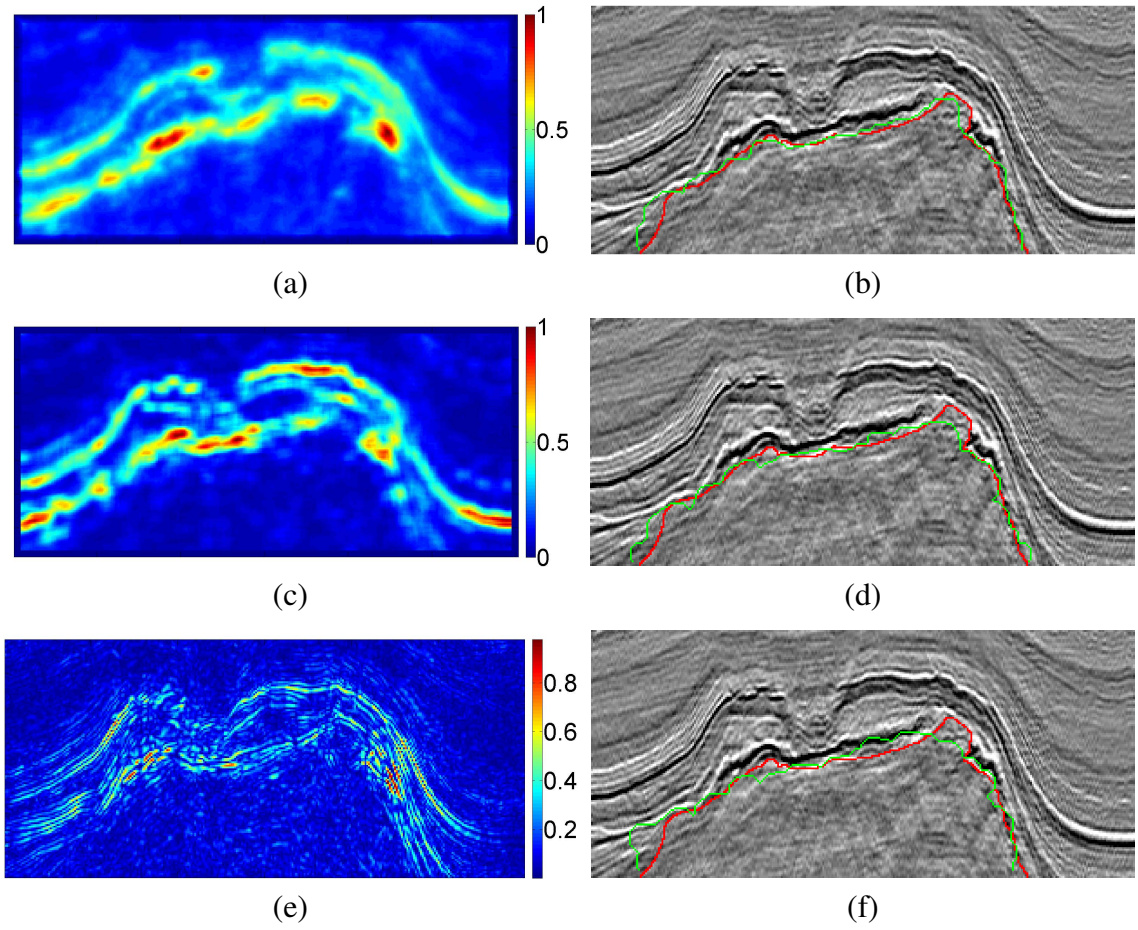
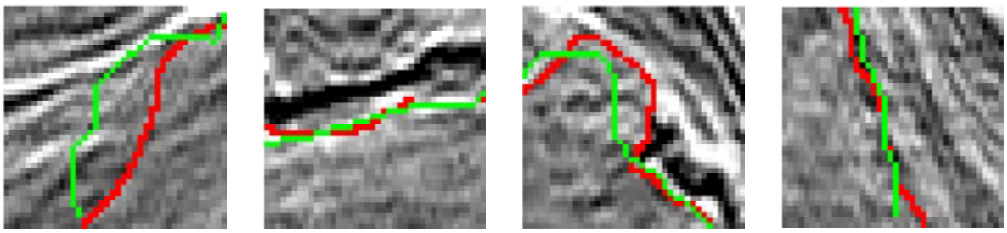


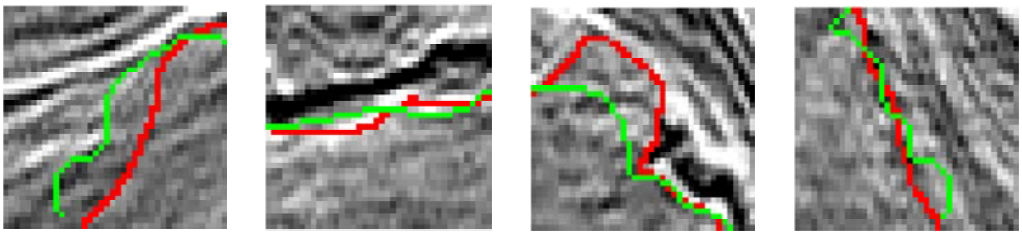
Figure 4.11: (a), (c), and (e) illustrate the GoT, GLCM contrast, and gradient maps of Inline #400, respectively; (b), (d), and (f) compare the manually labeled red ground truth with the green salt dome boundaries detected from attribute maps, (a), (c), and (e), respectively.

higher contrast values indicate likely boundary areas. By applying the region-growing-based detection workflow to Figure 4.11(c) , we can label the salt dome boundary as Figure 4.11(d) shows. Similar to the GLCM contrast attribute, the gradient attribute estimated based on 3D Sobel filter can also identify the boundaries of salt domes. To approximate the partial derivatives of different directions, we convolute three $3 \times 3 \times 3$ kernels of the Sobel filter with the local volume. For each point, the magnitude of the gradient attribute is calculated as follows:

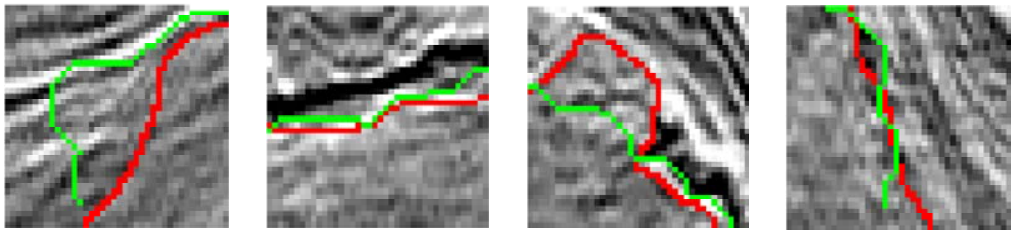
$$F = \sqrt{F_x^2 + F_y^2 + F_z^2}, \tag{4.28}$$



(a) Local boundaries extracted from the GoT map



(b) Local boundaries extracted from the GLCM contrast map



(c) Local boundaries extracted from the gradient map

Figure 4.12: Local salt-dome boundaries extracted from the GoT, GLCM contrast, and gradient maps are labeled in green. The ground truth labeled manually is labeled in red.

where F_x, F_y, and F_z represent the partial derivatives of the crossline, depth, and inline directions, respectively. Figure 4.11(e) illustrates the gradient map of Inline #400, from which we extract the green salt dome boundary as Figure 4.11(f) depicts.

By comparing the GoT and GLCM contrast of Inline #400 in Figure 4.11, we notice that it is not easy to determine which one leads to the more accurate detection of salt dome boundaries because of their comparable performance on highlighting boundary areas. However, without multiscale-based description, the gradient map depending only on 3×3 local regions shows noisy and discontinuous stripes around boundary areas, which limits the accuracy of detected boundaries. Figure 4.12 illustrates the local regions of Figures 4.11(b), (d), and (f), in which every column contains the same local regions of salt-dome boundaries detected from various attribute maps and every row contain the different

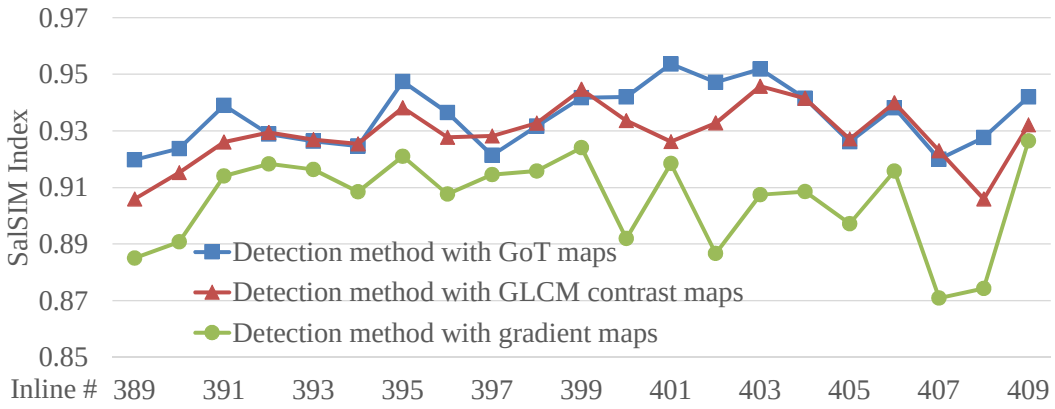


Figure 4.13: The SalSIM indices of salt-dome boundaries detected from the GoT, GLCM contrast, and gradient maps.

local regions of a detected salt-dome boundary. The first column illustrates the bottom-left corner of detected boundaries, in which the second one is slightly more similar to the ground truth than the other two local boundaries. However, in second to fourth columns, boundaries detected from the multi-scale GoT map show the highest accuracy.

Objective Comparison of Detected Boundaries

To quantize subjective perception, we employ the SalSIM index to evaluate the similarity between the ground truth and the detected boundary. Figure 4.13 shows the SalSIM indices of the salt-dome boundaries of Inline #389 to #409 detected by various methods. Boundaries extracted from gradient maps have the lowest accuracy because of noise around boundary areas. In most of the tested seismic sections, boundaries detected from GoT maps have SalSIM indices higher than or equal to those obtained from GLCM contrast maps. Since the computation of GLCMs is time consuming, the GoT-based detection method outperforms the GLCM-based one on both efficiency and accuracy. Table 4.1 lists several statistical measures of SalSIM indices, in which the averaged maximum distance (AMD) in pixels represents the mean of the d_{max} of all detected boundaries. The SalSIM indices of boundaries detected from GoT maps have the largest mean, the smallest standard deviation, and the shortest AMD, which proves the superiority of the GoT-based detection method.

Table 4.1: The statistical measures of SalSIM indices in Figure 4.13 obtained from various detection methods.

Detection Methods	Mean	Standard Deviation	AMD (pixels)
Detection method based on GoT Maps	**0.9348**	**0.0104**	**11.64**
Detection method based on GLCM Contrast Maps [30]	0.9290	**0.0104**	13.92
Detection method based on Gradient Maps [99]	0.9054	0.0159	20.73

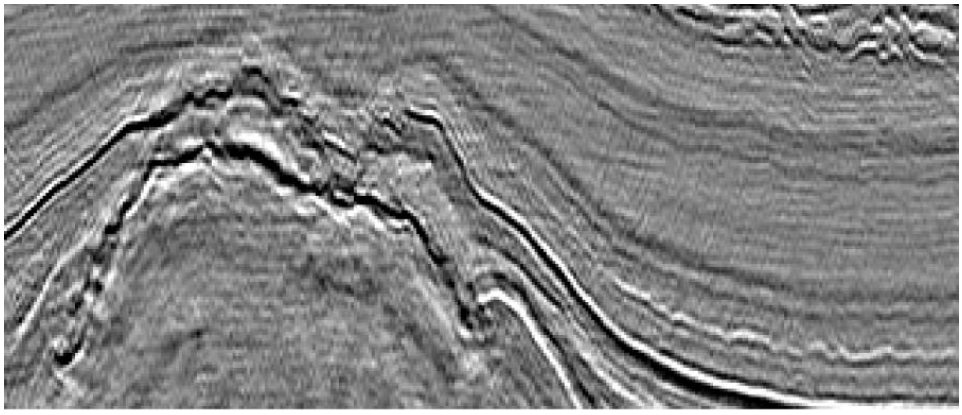


Figure 4.14: The SalSIM indices of salt-dome boundaries detected from the GoT, GLCM contrast, and gradient maps.

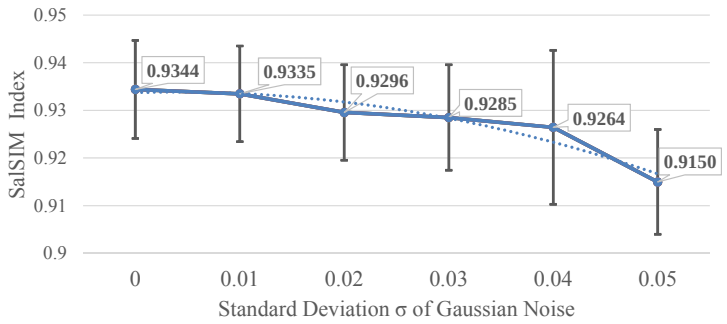
To further verify the robustness of the GoT-based detection method, we apply it on the ten sections of the F3 block with crossline number ranging from #834 to #843, inline number ranging from #279 to #600, and the time interval between 1,300ms and 1,848ms sampled every 4ms. Figure 4.14 illustrates one section of this local dataset, and Table 4.2 shows the performance of various detection methods, in which the GoT-based detection method still achieves the highest accuracy.

Robustness to Noise

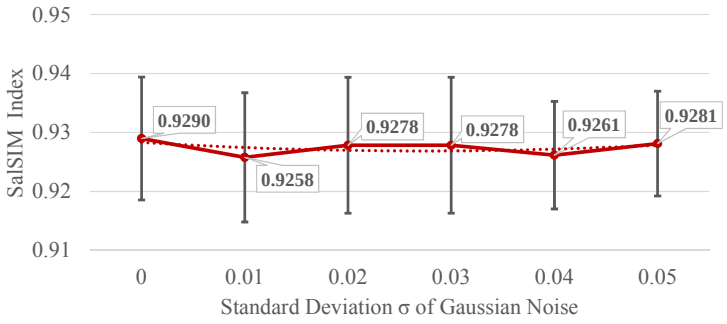
To verify the robustness of the GoT-based detection method, we add Gaussian noise to seismic sections and compare the accuracy of salt-dome boundaries detected from the corresponding GoT and GLCM contrast maps. The added zero-mean Gaussian noise has

Table 4.2: The statistical measures of the SalSIM indices of boundaries delineated by various detection methods in the crossline dataset.

Detection Methods	Mean	Standard Deviation	AMD (pixels)
Detection method based on GoT Maps	**0.9684**	**0.00280**	**4.67**
Detection method based on GLCM Contrast Maps [30]	0.9645	0.00379	5.68
Detection method based on Gradient Maps [99]	0.9396	0.00860	9.98



(a)The GoT-based detection method



(b)The GLCM-based detection method

Figure 4.15: The averaged SalSIM indices of salt-dome boundaries detected from noisy seismic sections using GoT- and GLCM-based detection methods.

standard deviations ranging from 0.01 to 0.05, and Figure 4.15 illustrates the change of SalSIM indices according to the increasing of standard deviations. For one noise level, we extract the salt-dome boundaries from noisy seismic sections using either the GoT- or GLCM-based detection method, and the mean of corresponding SalSIM indices is shown in Figure 4.15. The error bars of means correspond to an uncertainty equal to one standard deviation, and the dashed curve represents the trend of means. Although the averaged

accuracy of salt-dome boundaries detected by the GoT-based detection method decreases slowly with the increasing of standard deviations, it is still greater than those of boundaries detected by the GLCM-based method at the most of noisy levels. We notice that with the increasing of noise the latter fluctuates around a certain SalSIM index rather than keep decreasing, the reason for which is that the quantization step in the calculation of the GLCM, functioning as a built-in noise filter, can weaken the influence of noise on the GLCM contrast maps. To fairly compare the robustness of these two methods, we apply an edge-preserving smoothing filter, the bilateral filter [127], to noisy seismic sections. We notice that with the extra denoising operation, the GoT-based method shows higher accuracy than the GLCM-based one. If the noise level keeps increasing, depending only on the denoising operation we may not be able to label satisfied boundaries from migrated seismic sections. In such cases, geophysicists commonly prefer to regenerate migrated seismic sections with higher quality from raw seismic data by reducing noise that appears in every preprocessing step [128].

Analysis of Computational Complexity

To analyze computational complexity of seismic attributes mentioned above, we denote the size of images, the size of the largest neighborhoods, and the number of quantized levels in the computation of GLCMs as $M \times M$, $N \times N$, and L, respectively. In the GoT-based detection method, every point corresponds to a group of neighborhoods, the size of which ranges from 3×3 to $N \times N$ with edge length increased by 2. Therefore, the number of different types of neighborhoods is $(N-3)/2+1 = (N-1)/2$. Based on functions in Eq. (4.16), the calculation of one GoT value has computational complexity $P(N)$ analyzed

as follows:

$$\begin{aligned} P(N) &= \frac{(N-1)}{2} \cdot (\underbrace{2N^2}_{abs(W-U)} + \underbrace{2\left(N^2 \log N^2 + N^2\right)}_{\text{twice } |\mathscr{F}\{\cdot\}|} + \underbrace{2N^2}_{\text{average}}) \\ &= (N-1) \cdot \left(\frac{7}{2}N^2 + 2N^2 \log N\right) = \mathcal{O}\left(N^3 \log N\right). \end{aligned} \tag{4.29}$$

According to Eq. (4.29), we obtain the total computational complexity as $\mathcal{O}\left(M^2 \cdot N^3 \log N\right)$.

The computational complexity of deriving GLCM contrast maps depends mainly on the complexity required to obtain the elements of GLCMs. To identify the value of point (i, j) in GLCMs, we need to search point pairs at a predefined offset, the average complexity of which is $\mathcal{O}(N^2)$. Since we commonly define $4 \cdot \lfloor \frac{N}{2} \rfloor$ offsets, for one point, the computational complexity of calculating its GLCM contrast value is shown as follows:

$$P(N, L) = \underbrace{4 \cdot \lfloor \frac{N}{2} \rfloor}_{\substack{\text{number of} \\ \text{offsets}}} \cdot (\underbrace{N^2 \cdot L^2}_{\substack{\text{all elements in} \\ \text{one GLCM}}} + \underbrace{N^2}_{\substack{\text{derive contrast} \\ \text{attribute}}}). \tag{4.30}$$

Therefore, by traversing all points in a seismic section, we obtain a GLCM contrast map with the total computational complexity, $\mathcal{O}\left(M^2 \cdot N^3 \cdot L^2\right)$. If the dimensions of L and N are similar, the complexity can be approximated by $\mathcal{O}\left(M^2 \cdot N^5\right)$. The gradient maps are the convolution of seismic sections and 3D Sobel filters, the separability of which reduces computational complexity to $\mathcal{O}(M^2)$.

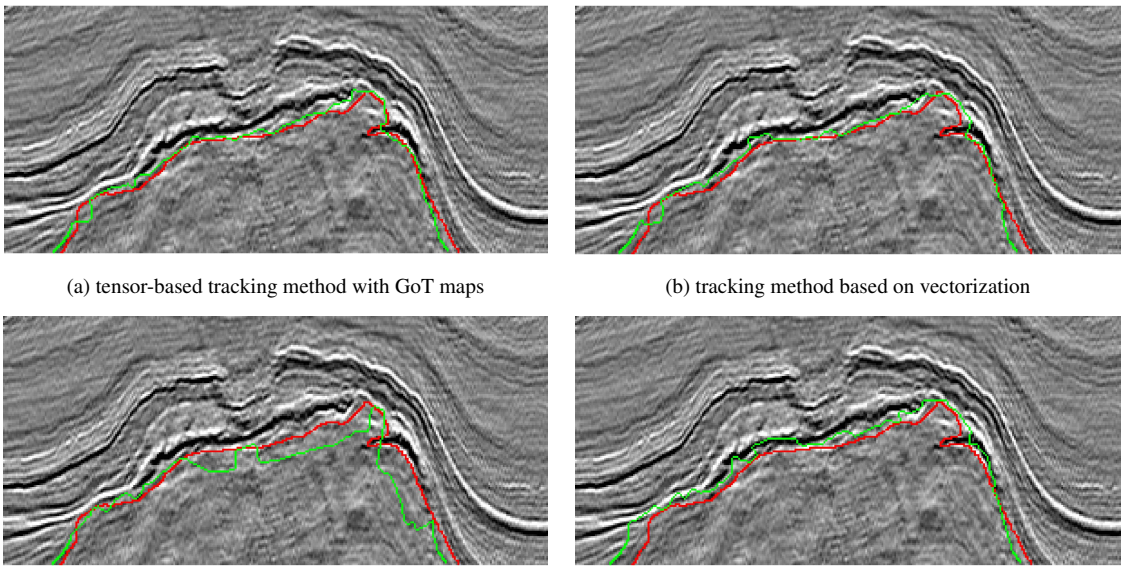
4.4.2 Salt-dome Tracking

In the local seismic volume we define Inline #400 as a reference section, the salt-dome boundary of which has been labeled by experienced interpreters or computer-aided detection methods. We can track the reference boundaries through the seismic volume and synthesize salt-dome boundaries in the neighboring twenty predicted sections ranging from Inline #389 to #409. Each point at the reference boundary corresponds to a pair of 31×31

image patches extracted from the reference section and its corresponding GoT map. To acquire texture features from the local regions of the reference boundary, we group all these patch pairs into tensor pairs on the basis of the block diagram shown in Figure 4.6. The dimensions of texture features in three modes are $[15, 15, 5]$, and threshold T_e on the reconstruction error is 2.3. Furthermore, in predicted sections the tracking method searches along the normal direction of the projected point and identifies the position of the tracked point by comparing the similarity between the patch pairs of candidate points and tensor pairs built from the reference section. The localization of tracked points is implemented automatically, which improves interpretation efficiency. Finally, we remove noisy points in predicted sections with 2×2 median filters and connect remaining points to label the salt-dome boundary under the shape constraint of salt domes. Figure 4.16 (a) compares the green tracked salt-dome boundary in Inline #391, synthesized based on the manually labeled reference boundary in Inline #400, with the red ground truth. We notice that these two curves almost overlap except for several local regions.

Subjective Comparison of Tracking Methods

To prove that tracking accuracy can be increased by involving the GoT attribute and the texture features of all modes, we need to compare the tensor-based tracking method based on GoT maps with three other tracking strategies. The first one is to implement the tracking process depending only on texture features extracted from vectorized patches rather than those from third-order tensors. The second one is to synthesize tracked boundaries within the framework of the tensor-based tracking method, but without involving the GoT attribute. The third strategy is to utilize the tensor-based tracking method, but replace GoT maps with GLCM contrast ones. We rename these three tracking strategies as the tracking method based on vectorization, the tensor-based tracking method without GoT maps, and the tensor-based tracking method with GLCM contrast maps. Figures 4.16 (b) to (d) compare the red ground truth labeled manually with green salt-dome boundaries labeled



(a) tensor-based tracking method with GoT maps

(b) tracking method based on vectorization

(c) tensor-based tracking method without GoT maps

(d) tensor-based tracking method with GLCM contrast maps

Figure 4.16: (a), (b), (c), and (d) show the comparison between the red ground truth and the green tracked salt dome boundaries of Inline #391 synthesized by the tensor-based tracking method with GoT maps, the tracking method based on vectorization, the tensor-based tracking method without GoT maps, and the tensor-based tracking method with GLCM contrast maps , respectively.

by the three tracking strategies mentioned above, and the tracked boundary in Figure 4.16 (c) shows the greatest deviation from the ground truth. Figure 4.17 illustrates the local regions of Figures 4.16 (a) to (d), in which every column contains the same local regions of salt-dome boundaries synthesized based on various tracking methods and every row contains the different local regions of a tracked salt-dome boundary. By comparing the local regions of tracked boundaries with the ground truth, we notice the boundary in the first row synthesized by the tensor-based tracking method with GoT maps is the most similar to the ground truth.

Objective Comparison of Tracking Methods

To objectively verify our conclusion, we synthesize tracked boundaries in seismic sections ranging from Inline #389 to #409 using various tracking strategies and plot their SalSIM indices in Figure 4.18. The horizontal and vertical axes represent the inline number and the

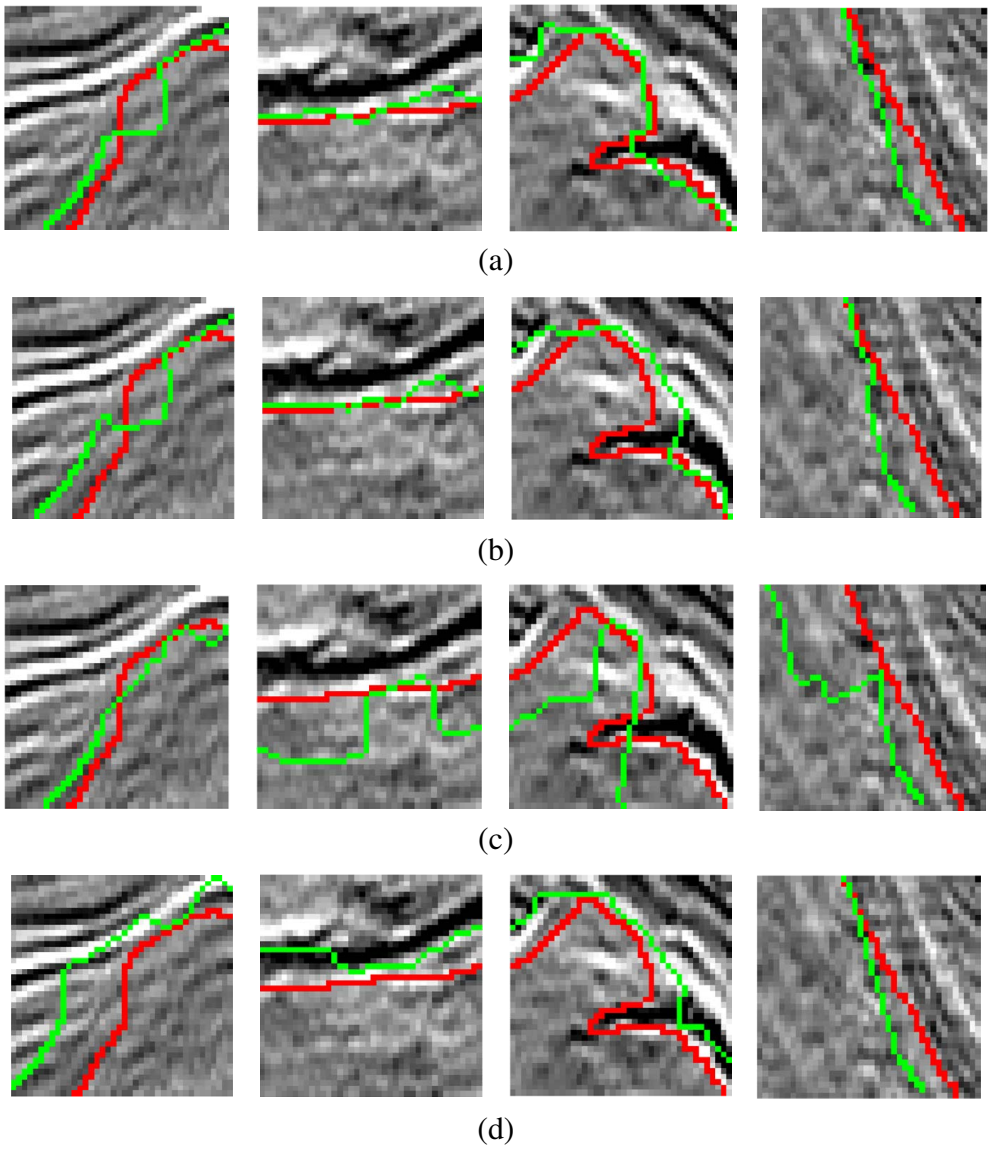


Figure 4.17: (a), (b), (c), and (d) contain four local regions extracted from Figures 4.16(a) to (d), respectively. The red and green curves represent tracked salt-dome boundaries and the ground truth, respectively.

SalSIM index, respectively. Although tracking method based on vectorization can achieve comparable accuracy with the tensor-based tracking method based on GoT maps in Inline #401 to #409, the SalSIM indices of salt-dome boundaries in Inline #395 to #389 synthesized by the former one decrease more quickly than those obtained by the latter one. In addition, dashed curves in Figure 4.18 shows the distribution trends of SalSIM indices. Although the SalSIM indices of four tracking strategies have similar trends that drop with the

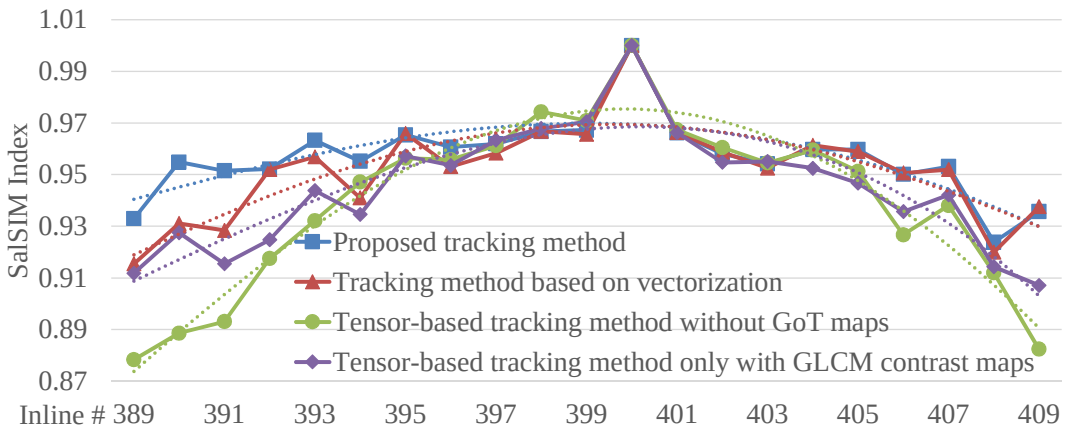
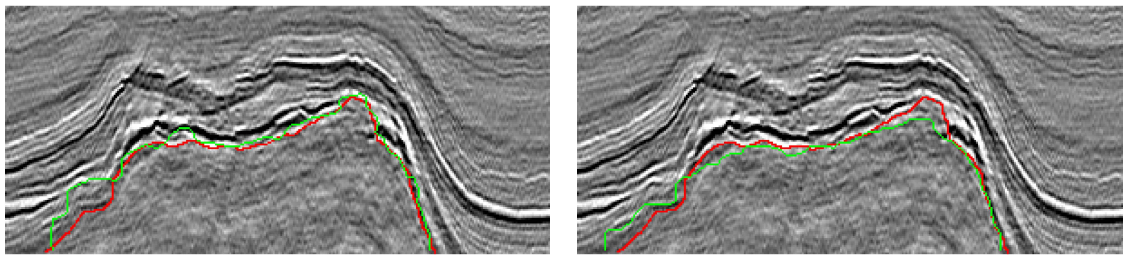


Figure 4.18: The SalSIM indices of tracked boundaries in predicted sections synthesized based on different tracking strategies.

increasing offsets between predicted sections and the reference section, the tensor-based tracking method with GoT maps outperforms other tracking strategies particularly in Inline #389 to #399. Table 4.3 contains several statistical measures of SalSIM indices in Figure 4.18. The mean of SalSIM indices and the AMD evaluate the accuracy of the tracking method, and the standard deviation determines robustness. The tensor-based tracking method with GoT maps has the greatest mean, the smallest standard deviation, and the shortest AMD, which proves its superiority on salt-dome tracking.

Table 4.3: The statistical measures of SalSIM indices in Figure 4.18.

Tracking Methods	Mean	Standard Deviation	AMD (pixels)
Tensor-based tracking method with GoT maps	**0.9531**	**0.0111**	**8.32**
Tracking method based on vectorization	0.9496	0.0148	9.30
Tensor-based tracking method without GoT maps	0.9364	0.0295	11.26
Tensor-based tracking method with GLCM contrast maps	0.9422	0.0189	10.53



(c) (d)

Figure 4.19: (a) tracked boundary of Inline #408 and (d) detected boundary of Inline #408.

Combination of Detection and Tracking Methods

In the previous sections, tracked boundaries are synthesized based on the boundary labeled by interpreters in the reference section. To further reduce human intervention in the tracking process, in the local seismic volume, we can synthesize the tracked boundaries of Inline #389 to #409 on the basis of the reference boundary in Inline #400 labeled by the detection method. Figure 4.19 compares the tracked and detected green boundaries with the red ground truth. In Figures 4.19 (c) and (d), since the bottom-left corner of the tracked boundary has great deviation to the ground truth, the detected boundary in Inline #408 seems to be more accurate. However, the appearance of the great deviation is caused by the deviation in the reference boundary shown in Figure 4.11 (b) rather than our tracking method. To objectively compare the tracked and detected boundaries, we illustrate the corresponding SalSIM indices in Figure 4.20. Since we employ λ_c as an important weight that helps move tracked points to the boundary area, the tracked boundaries may have higher SalSIM indices than the detected ones, particularly in Inline #395 to #399. The mean of the SalSIM indices of the tracked boundaries is 0.9362, which is almost the same as those of the detected boundaries, 0.9348. It shows that the tensor-based tracking method with GoT maps is robust and efficient enough to delineate salt dome boundaries.

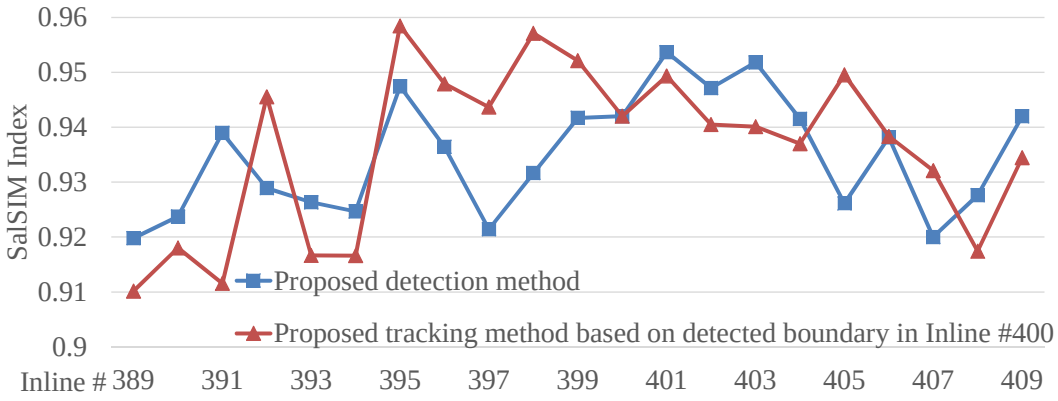


Figure 4.20: The SalSIM indices of tracked boundaries in predicted sections synthesized based on different tracking strategies.

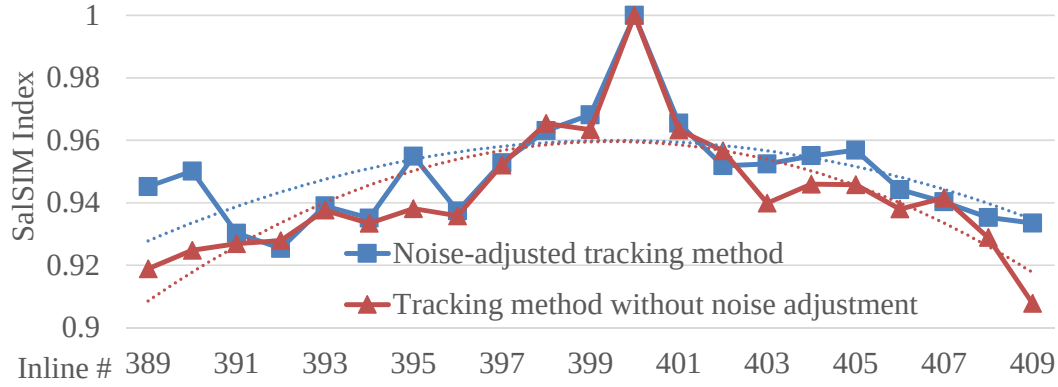


Figure 4.21: The comparison of boundaries in predicted sections synthesized by the tensor-based tracking method with or without noise adjusted.

Influence of Noise on Tracking Method

Since in some cases the noise level in the reference section may be different from those of predicted sections, to eliminate the influence of noise on the tracked process, we introduce the noise-adjusted tracking method, which selects PCs corresponding to the greatest SNR other than the greatest variation. In the local volume, we manually add gaussian noise with different variances to seismic sections. In the reference section Inline #400, the added zero-mean gaussian noise has a variance of 0.01. In contrast, the noise level increases to 0.05 in predicted sections. We utilize the manually labeled reference boundary to synthesize tracked boundaries in Inline #389 to #409. In Figure 4.21, We compare boundaries synthesized by the tensor-based tracking method with or without noise adjusted. With the

increasing offset between the reference and predicted sections, the appearance of noise magnifies the error in the localization of tracked points. Since the noise-adjusted tracking method is robust to noise, the difference of two curves in Figure 4.21 grows with the increasing offset. The tracked boundaries obtained by the former one have a mean of SalSIM indices, 0.9468, which is greater than that of the boundaries synthesized by the latter one, 0.9396. It shows that the noise-adjusted tracking method can alleviate the influence of noise and enhance the robustness of the tracking process.

4.5 Summary

In the chapter, we extended the concept of the interactive interpretation framework introduced in Chapter 3 to the labeling of salt domes. In a seismic volume, we classified its seismic sections containing salt domes into reference and predicted ones, and applied different strategies for these two types of sections. Since salt domes containing chaotic reflections have different texture from surrounding structures formed by other types of rocks, to capture the change of texture between salt and non-salt regions, we calculated the GoT attribute of all sections by involving a perceptual dissimilarity measure. In reference sections, we highlighted salt-dome boundaries with higher GoT values using a threshold automatically generated from Ostu's method. From an interactively selected initialization point inside the salt dome, we performed region growing, and regions indicating the salt dome keep growing and stop at boundary regions. After the closing operation, grown regions determine the boundaries of the salt dome in reference sections. By tracking detected boundaries in reference sections that constitute the minority of seismic sections through a seismic volume, we further improve the efficiency of salt-dome interpretation. Since local areas along salt-dome boundaries commonly have similar textures, we built texture tensors by stacking texture patches around salt-dome boundaries in the reference section along the third mode. While building tensors, we introduced the incremental tensor-based PCA to implement tensor classification. Although seismic data commonly contain noise, noise-

adjusted tensor-based PCA generated a set of transform matrices by selecting PCs with the highest SNR. To localized boundary points in predicted sections, we first projected the detected boundary in the reference section onto the target predicted section and kept the coordinates of all points unchanged. Then, to identify the tracked position of every projected point, we searched among candidate points located along the normal direction of the projected boundary. Every candidate point corresponded to a patch pair, which could be reconstructed by obtained transform matrices. The candidate point with the smallest reconstruction error is selected as the tracked point. To evaluate the performance of the interactive framework on salt-dome interpretation, we introduced the salt-dome similarity (SalSIM) index that described the similarity between detected and manually picked salt-dome boundaries. The SalSIM index based on the Fréchet distance compares both the local and global structures of salt-dome boundaries. Experimental results showed that the interactive framework has the capability to accurately detect salt-domes in the seismic volume, and the objective SalSIM indices complied with the subjective observation of interpreters.

CHAPTER 5

TENSOR-BASED SUBSPACE LEARNING AND ITS APPLICATION IN SALT-DOME TRACKING

According to our discussion in Chapter 4, an effective texture descriptor or texture feature extraction method can help improve the efficiency and accuracy of salt-dome interpretation. However, in Chapter 4, both the GoT attribute and the texture tensor are generated from 2D sections, which fails to capture texture correlations between neighboring sections. Such a shortcoming in most salt-dome interpretation methods has become a limit on interpretation performance. To solve this problem, in this chapter we introduce a tensor-based subspace learning method that extracts features from multidimensional data and apply it on the interactive interpretation framework for extracting texture features in salt-dome tracking. In the field of signal processing, tensors represent multidimensional arrays and subspace learning is an important approach for dimensionality reduction. Therefore, before we introduce the novel tensor-based subspace learning algorithm, we will first introduce the background of multilinear dimensionality reduction and a conventional tensor-based subspace learning method.

5.1 Multilinear Dimensionality Reduction

5.1.1 Multilinear Dimensionality Reduction Model

Suppose that $\left\{\boldsymbol{\mathcal{X}}_m \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}, m = 1, 2, \cdots, M\right\}$ represents a data set containing M tensor samples residing in tensor space $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \otimes \cdots \otimes \mathbb{R}^{I_n}$. The general purpose of multilinear dimensionality reduction is to find a multilinear transformation $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times P_n}$, $P_n < I_n$, $n = 1, 2, \cdots, N$, that maps samples in space $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \otimes \cdots \otimes \mathbb{R}^{I_N}$ to space $\mathbb{R}^{P_1} \otimes \mathbb{R}^{P_2} \otimes \cdots \otimes \mathbb{R}^{P_N}$. Since $\mathbf{U}^{(n)} = \left[\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, \cdots, \mathbf{u}_{P_n}^{(n)}\right]$ consists of a set of basis vectors

in the P_n-dimensional space, the mapping process related to multilinear dimensionality reduction can be expressed as follows:

$$\boldsymbol{\mathcal{Y}}_m = \boldsymbol{\mathcal{X}}_m \times_1 \mathbf{U}^{(1)^T} \times_2 \mathbf{U}^{(2)^T} \cdots \times_N \mathbf{U}^{(N)^T}, m = 1, 2, \cdots, M. \tag{5.1}$$

Mapped samples $\left\{\boldsymbol{\mathcal{Y}}_m \in \mathbb{R}^{P_1 \times P_2 \times \cdots \times P_N}, m = 1, 2, \cdots, M\right\}$ in a space with fewer dimensions represent features extracted from $\{\boldsymbol{\mathcal{X}}_m, m = 1, 2, \cdots, M\}$ under certain optimal criteria. To identify transformation matrices $\left\{\mathbf{U}^{(1)}, \mathbf{U}^2, \cdots, \mathbf{U}^{(N)}\right\}$, different criteria were involved in different algorithms.

5.1.2 Tensor Linearity Projection Preserving (TLPP)

Tensor linearity projection preserving (TLPP) as an effective multilinear dimensionality reduction method keeps the local structure of tensor samples. To reveal local neighborhood relationships between samples, TLPP constructs adjacent graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where nodes $\mathcal{V}$ are tensor samples and edges $\mathcal{E}$ are associated with adjacent samples. In the process that maps tensor samples to a lower dimensional space, TLPP preserves the graph structure by minimizing the objective function as follows:

$$\min_{\left\{\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \cdots, \mathbf{u}^{(N)}\right\}} \frac{1}{2} \sum_{i,j}^{M} (y_i - y_j)^2 \mathbf{W}_{ij}, \tag{5.2}$$

where $\left\{\mathbf{u}^{(n)} \in \mathbb{R}^{I_n \times 1}, n = 1, 2, \cdots, N\right\}$ are a set of transformation vectors for N modes. $y_i = \boldsymbol{\mathcal{X}}_i \times_1 \mathbf{u}^{(1)^T} \times_2 \mathbf{u}^{(2)^T} \cdots \times_N \mathbf{u}^{(N)^T}, i = 1, 2, \cdots, M$, mapped from sample $\boldsymbol{\mathcal{X}}_i$ represents a tensor, each mode of which has only one dimension. In addition, $\mathbf{W}_{ij}$ represents the weight of the edge associated with samples $\boldsymbol{\mathcal{X}}_i$ and $\boldsymbol{\mathcal{X}}_j$ and has the value defined as follows:

$$\mathbf{W}_{ij} = \begin{cases} K(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{X}}_j), & \boldsymbol{\mathcal{X}}_i \in \mathcal{N}(\boldsymbol{\mathcal{X}}_j) \text{ or } \boldsymbol{\mathcal{X}}_j \in \mathcal{N}(\boldsymbol{\mathcal{X}}_i), \\ 0, & \text{Otherwise}, \end{cases} \tag{5.3}$$

where $\mathcal{N}(\cdot)$ denotes the adjacency list of one sample. In unsupervised learning, the adjacency list of sample $\boldsymbol{\mathcal{X}}_i$ may contain its k-nearest neighbors or neighbors within radius ϵ. In supervised learning, the adjacency list of sample $\boldsymbol{\mathcal{X}}_i$ commonly contains all samples having the same label with $\boldsymbol{\mathcal{X}}_i$. In Eq. (5.3), weight function $K(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{X}}_j)$ measures the similarity between $\boldsymbol{\mathcal{X}}_i$ and $\boldsymbol{\mathcal{X}}_j$, and one common choice for the weight function is the Gaussian kernel, $K\left(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{X}}_j\right) = \exp\left(-\left\|\boldsymbol{\mathcal{X}}_i - \boldsymbol{\mathcal{X}}_j\right\|_F^2 / \sigma\right)$, where σ is set empirically. Therefore, weights in $\mathbf{W}$ are nonzero only for adjacent nodes.

To obtain the matrix representation of the objective function in Eq. (5.2), we denote y_i in a way similar to the matrix representation of the HOSVD in Eq. (4.15) as follows:

$$\begin{aligned} y_i &= \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \\ &= \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \left(\mathbf{u}^{(n+1)} \otimes \cdots \otimes \mathbf{u}^{(N)} \otimes \mathbf{u}^{(1)} \otimes \cdots \otimes \mathbf{u}^{(n-1)}\right), \end{aligned} \tag{5.4}$$

where $\mathbf{X}_{i(n)}$ is the n-mode unfolding matrix of $\boldsymbol{\mathcal{X}}_i$. With the expression of y_i, we derive the matrix representation of the objective function in Eq. (5.2) as follows:

$$\begin{aligned} &\frac{1}{2}\sum_{i,j}^{M} \left(y_i - y_j\right)^2 \mathbf{W}_{ij} \\ =&\frac{1}{2}\sum_{i,j}^{M} \left(\mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} - \mathbf{u}^{(n)^T} \mathbf{X}_{j(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}\right) \left(\mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} - \mathbf{u}^{(n)^T} \mathbf{X}_{j(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}\right)^T \mathbf{W}_{ij} \\ =&\sum_{i,j}^{M} \mathbf{W}_{ij} \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}^T \mathbf{X}_{i(n)}^T \mathbf{u}^{(n)} - \sum_{i,j}^{M} \mathbf{W}_{ij} \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}^T \mathbf{X}_{j(n)}^T \mathbf{u}^{(n)} \\ =&\sum_{i}^{M} \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \mathbf{D}_{ii} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}^T \mathbf{X}_{i(n)}^T \mathbf{u}^{(n)} - \sum_{i,j}^{M} \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \mathbf{W}_{ij} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}^T \mathbf{X}_{j(n)}^T \mathbf{u}^{(n)} \\ =&\mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left(\mathbf{I}_M \otimes \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}\right) \left(\mathbf{D} - \mathbf{W}\right) \left(\mathbf{I}_M \otimes \mathbf{u}_{\boldsymbol{\Phi}^{(n)}}^T\right) \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}, \end{aligned} \tag{5.5}$$

where $\mathbf{X}_{(n)} = \left[\mathbf{X}_{1(n)}, \mathbf{X}_{2(n)}, \cdots, \mathbf{X}_{M(n)}\right]$ contains the n-mode unfolding matrices of all samples. $\mathbf{D}$ is a diagonal matrix, the entries of which are the row sums of $\mathbf{W}$, $\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{W}_{ij}$. In addition, we define $\mathbf{U}_{\boldsymbol{\Phi}_{(n)}}^{\mathbf{I}_M} = \mathbf{I}_M \otimes \mathbf{u}_{\boldsymbol{\Phi}_{(n)}}$, where matrix $\mathbf{I}_M$ represents an

identity matrix with a size of M. The transpose of matrix $\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}$ is denoted as $\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}$. Eq. (5.5) is minimized subject to constraint $\mathbf{y}\mathbf{D}\mathbf{y}^T = 1$, where $\mathbf{y} = [y_1, y_2, \cdots, y_M]$ and $\mathbf{D}$ measures the importance of y_i. Using notations in Eq. (5.5), we convert the constraint to

$$\mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\mathbf{D}\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}\mathbf{X}_{(n)}^T\mathbf{u}^{(n)} = 1. \tag{5.6}$$

With the assumption that $\left\{\mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(n-1)}, \mathbf{u}^{(n+1)}, \cdots, \mathbf{u}^{(N)}\right\}$ are known, by finding optimal $\mathbf{u}^{(n)}$, we minimize the objective function in Eq. (5.2) with the constraint in Eq. (5.6) as follows:

$$\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\left(\mathbf{D}-\mathbf{W}\right)\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}\mathbf{X}_{(n)}^T\mathbf{u}^{(n)} \text{ s.t. } \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\mathbf{D}\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}\mathbf{X}_{(n)}^T\mathbf{u}^{(n)} = 1. \tag{5.7}$$

Transformation vector $\mathbf{u}^{(n)}$ that minimizes the objective function is given by the minimum eigenvalue solution to the generalized eigenvalue problem as follows:

$$\left(\mathbf{X}_{(n)}\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\mathbf{L}\cdot\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}\mathbf{X}_{(n)}^T\right)\mathbf{v} = \lambda\left(\mathbf{X}_{(n)}\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\mathbf{W}\cdot\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M}\mathbf{X}^T\right)\mathbf{v}, \tag{5.8}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the Laplacian matrix. The eigenvectors of Eq. (5.8) corresponding to the P_n smallest eigenvalues constitute transformation matrix $\mathbf{U}^{(n)}$. In a similar iterative manner, we can obtain transform matrices $\left\{\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}\right\}$ by solving corresponding generalized eigenvalue problems.

From Eq. (5.2), we summary the two drawbacks of TLPP:

(1) Since TLPP preserves the topology structure of tensor samples using the adjacent graph, neighboring samples will be embedded as closely as possible in the tensor subspace. The mapping process of TLPP involves only the similarity between neighboring samples, but ignores their variability. It means that the relative location relationships of samples may be lost in the embedded tensor subspace. To illustrate this drawback of TLPP, in Figure 5.1 we plot four two-dimensional (2D) neighboring samples and

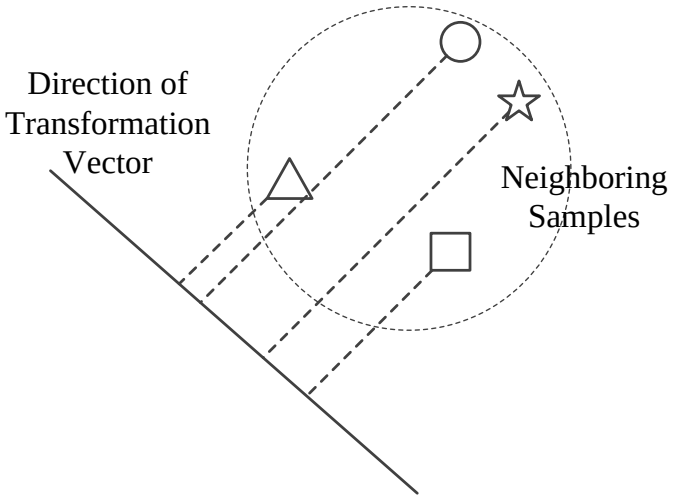


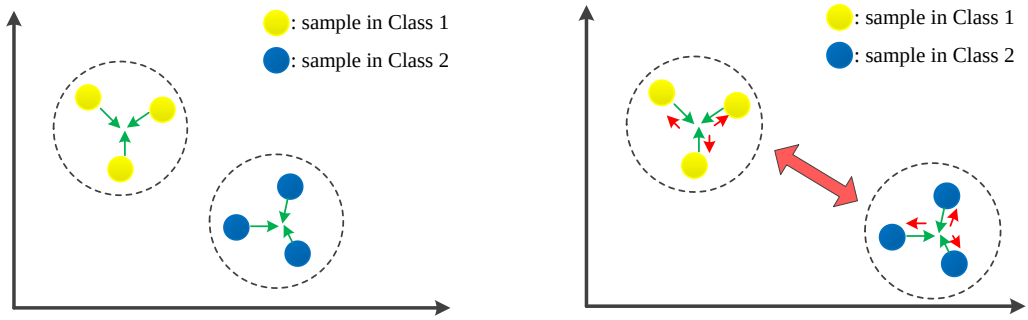
Figure 5.1: The projection of neighboring samples on the transformation vector.

the direction of the corresponding transformation vector. We notice that sample pair $\{\triangle, \bigcirc\}$ having the largest distance in the 2D space become the nearest one in the embedded one-dimensional (1D) space compared to other embedded sample pairs. The missing of relative location relationships may degrade the performance of TLPP on classification and retrieval.

(2) TLPP is able to embed neighboring tensor samples together in the subspace by preserving their topology structures. However, the preserving of local structures in TLPP does not fully utilize the discriminant structures of all samples. For example, in supervised learning, TLPP preserves the local structures of samples in the same class, but fails to encode discriminant information between different classes. Therefore, classes containing similar samples may not be well distinguished in TLPP. The lack of global discriminant information may degrade the recognition performance of TLPP.

5.2 Tensor Orthogonal Locality Discriminant Projection with Maximum Margin Criterion (TOLDP-MMC)

To overcomes the drawbacks of TLPP, we introduce a novel tensor-based subspace learning method, tensor orthogonal locality discriminant projection with maximum margin criterion



(a) TLPP (b) OLDPMMC

Figure 5.2: Comparison between TLPP and TOLDP-MMC.

(TOLDP-MMC). Figure 5.2 illustrates the difference between TLPP and TOLDP-MMC. TLPP embeds neighboring samples in the subspace as closely as possible, which is indicated by green arrows in Figure 5.2(a). In contrast, the TOLDP-MMC preserves both the similarity and variability of neighboring samples, which are illustrated by green and small red arrows in Figure 5.2(a), respectively. In addition, the TOLDP-MMC can also keep the global structure of all samples with discriminant information involved, as the red large arrow in Figure 5.2(b) shows. TOLDP-MMC contains three main parts, the tensor locality-preserved diversity projection, the maximum margin criterion, and orthogonality.

5.2.1 Tensor Locality-preserved Diversity Projection

Similar to TLPP, the tensor locality-preserved diversity projection (TLDP) first defines adjacent graph $\mathcal{G}_{\mathcal{S}} = \{\mathcal{V}_{\mathcal{S}}, \mathcal{E}_{\mathcal{S}}\}$, which describes the neighboring relationships of samples. In graph $\mathcal{G}_{\mathcal{S}}$, edges $\mathcal{E}_{\mathcal{S}}$ associated with adjacent samples have weights in matrix $\mathbf{W_S}$, which is same to $\mathbf{W}$ in Eq. (5.3). To keep the similarity of tensor samples in the embedded subspace, we minimize the objective function same to Eq. (5.2) as follow:

$$\min_{\left\{\mathbf{u}^{(1)},\mathbf{u}^{(2)},\cdots,\mathbf{u}^{(N)}\right\}} \frac{1}{2}\sum_{i,j}^{M}\left(y_i - y_j\right)^2 \mathbf{W}_{\mathbf{S}ij}. \tag{5.9}$$

By assuming that $\left\{\mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(n-1)}, \mathbf{u}^{(n+1)}, \cdots, \mathbf{u}^{(N)}\right\}$ are known, we can obtain the simplified objective function same to Eq. (5.5) as follows:

$$\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}_{(n)}} \mathbf{L}_{\mathbf{S}} \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}^T_{(n)}} \mathbf{X}^T_{(n)} \mathbf{u}^{(n)}, \tag{5.10}$$

where Laplacian matrix $\mathbf{L_S} = \mathbf{D_S} - \mathbf{W_S}$. Eq. (5.10) ensures that neighboring samples in the same class will be embedded as closely as possible in the subspace.

The diversity of samples, as another criterion, indicates samples' relative position information. A pair of samples with a large distance will still has larger distance in the embedded subspace compared to other mapped sample pairs. To describe the diversity of samples, we define another adjacent graph $\mathcal{G}_\mathcal{R} = \{\mathcal{V}_\mathcal{R}, \mathcal{E}_\mathcal{R}\}$. $\mathcal{G}_\mathcal{R}$ is same to $\mathcal{G}_\mathcal{S}$ except for weight matrix $\mathbf{W_R}$, the entries of which are calculated as follows:

$$\mathbf{W}_{\mathbf{R}ij} = \begin{cases} 1 - K(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{X}}_j), & \boldsymbol{\mathcal{X}}_i \in \mathcal{N}(\boldsymbol{\mathcal{X}}_j) \text{ or } \boldsymbol{\mathcal{X}}_j \in \mathcal{N}(\boldsymbol{\mathcal{X}}_i), \\ 0, & \text{Otherwise.} \end{cases} \tag{5.11}$$

We assign larger weights to sample pairs with larger distances and preserve relative position relationships between samples by maximizing the objective function as follows:

$$\max_{\left\{\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \cdots, \mathbf{u}^{(N)}\right\}} \frac{1}{2} \sum_{i,j}^{M} \left(y_i - y_j\right)^2 \mathbf{W}_{\mathbf{R}ij}. \tag{5.12}$$

Under the assumption that $\left\{\mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(n-1)}, \mathbf{u}^{(n+1)}, \cdots, \ \mathbf{u}^{(N)}\right\}$ are known, we can obtain the simplified objective function as follows:

$$\max_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}_{(n)}} \mathbf{L}_{\mathbf{R}} \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}^T_{(n)}} \mathbf{X}^T_{(n)} \mathbf{u}^{(n)}, \tag{5.13}$$

where $\mathbf{L_R} = \mathbf{D_R} - \mathbf{W_R}$. To balance between the similarity and diversity of samples, we

combine objective functions in Eqs. (5.10) and (5.13) using parameter α as follows:

$$\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}_{(n)}} \left(\alpha \mathbf{L_S} - (1-\alpha)\, \mathbf{L_R}\right) \mathbf{U}^{\mathbf{I}_M}_{\mathbf{\Phi}^T_{(n)}} \mathbf{X}^T_{(n)} \mathbf{u}^{(n)}. \tag{5.14}$$

α emphasizes the importance of the compactness of samples in the embedded subspace and commonly ranges between 0.5 and 1. In our experiments, we set $\alpha = 0.9$. In contrast, the small weight on Eq. (5.13) helps preserve relative position relationships and avoid over-fitting that may be caused by Eq. (5.10). To understand Eq. (5.14), we re-write $\alpha \mathbf{L_S} - (1-\alpha)\, \mathbf{L_R}$ as Laplacian matrix $\mathbf{L_C} = \mathbf{D_C} - \mathbf{W_C}$ and weights in $\mathbf{W_C}$ are calculated as follows:

$$\mathbf{W_C}_{ij} = \begin{cases} K(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{X}}_j) + \alpha - 1, & \boldsymbol{\mathcal{X}}_i \in \mathcal{N}(\boldsymbol{\mathcal{X}}_j) \text{ or } \boldsymbol{\mathcal{X}}_j \in \mathcal{N}(\boldsymbol{\mathcal{X}}_i), \\ 0, & \text{Otherwise.} \end{cases} \tag{5.15}$$

The nature of the TLDP is that slightly decreased distances between samples can help avoid the over-fitting of TLPP.

5.2.2 Tensor Maximum Margin Criterion

In addition to criteria that preserves the similarity and diversity of neighboring samples, the TOLDP-MMC involves the tensor maximum margin criterion to encode global discriminant information. Similar to MMC, in the embedded subspace TMMC separates samples from different classes as far as possible by maximizing the difference between the within- and between-class variances of embedded samples. We commonly describe the within-class variance using the within-class scatter matrix. In supervised learning, we assume that M tensor samples belong to C classes and modify the notation of samples as $\boldsymbol{\mathcal{X}}_{c,i}$, where $c = 1, 2, \cdots, C$, indicates the class label and $i = 1, 2, \cdots, N_c$, represents the sample index in class c. The n-th mode unfolding matrix of $\boldsymbol{\mathcal{X}}_{c,i}$ is $\mathbf{X}_{c,i(n)}$. $\mathbf{X}_{c(n)} = \left[\mathbf{X}_{c,1(n)}, \mathbf{X}_{c,2(n)}, \cdots, \mathbf{X}_{c,N_c(n)}\right]$ concatenates all n-th mode unfolding matrices in

class c. In addition, we need to define three constant items, identity matrices $\mathbf{I}_{N_c}$ and $\mathbf{I}_{\Pi(n)}$ with the sizes of N_c and $\prod_{k\neq n} I_k$, respectively, and vector $\mathbf{e}_{N_c}$ with all N_c entries equal to one. On the basis of the definition of the within-class variance, we simplify the within-class scatter matrix of tensor samples as follows:

$$\begin{aligned}
\mathbf{S}_w &= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\left(y_i-\bar{y}_c\right)\left(y_i-\bar{y}_c\right)^T \\
&= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\mathbf{u}^{(n)^T}\left(\mathbf{X}_{c,i(n)}-\overline{\mathbf{X}}_{c(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{X}_{c,i(n)}^T-\overline{\mathbf{X}}_{c(n)}^T\right)\mathbf{u}^{(n)} \\
&= \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\cdot diag\left(\mathbf{L}_1,\mathbf{L}_2,\cdots,\mathbf{L}_C\right)\cdot\mathbf{X}_{(n)}^T\mathbf{u}^{(n)} \\
&= \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\cdot\mathbf{L}_{\mathbf{W}}\cdot\mathbf{X}_{(n)}^T\mathbf{u}^{(n)},
\end{aligned} \tag{5.16}$$

where $\mathbf{L}_c$, $c=1,2,\cdots,C$, is defined as

$$\mathbf{L}_c = \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_{N_c}}\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_{N_c}} - \frac{1}{N_c}\left(\mathbf{e}_{N_c}\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\Pi(n)}\right). \tag{5.17}$$

The detailed proof of Eq. (5.16) can be found in Appendix A.1. Since the total variance of samples is equal to the summation of within- and between-class variances, the between-class variance can represented by the difference of total scatter matrix $\mathbf{S}_t$ and within-class scatter matrix $\mathbf{S}_w$. According to the definition of the total variance, we simplify the total scatter matrix of tensor samples as follows:

$$\begin{aligned}
\mathbf{S}_t &= \sum_{m=1}^{M}\left(y_m-\bar{y}\right)\left(y_m-\bar{y}\right)^T \\
&= \sum_{m=1}^{M}\mathbf{u}^{(n)^T}\left(\mathbf{X}_{m(n)}-\overline{\mathbf{X}}_{(n)}\right)\mathbf{u}_{\mathbf{\Phi}(n)}\mathbf{u}_{\mathbf{\Phi}(n)}^T\left(\mathbf{X}_{m(n)}-\overline{\mathbf{X}}_{(n)}\right)^T\mathbf{u}^{(n)} \\
&= \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\left[\mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M}\cdot\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} - \frac{1}{M}\left(\mathbf{e}_M\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{u}_{\mathbf{\Phi}(n)}\mathbf{u}_{\mathbf{\Phi}(n)}^T\left(\mathbf{e}_M^T\otimes\mathbf{I}_{\Pi(n)}\right)\right]\mathbf{X}_{(n)}^T\mathbf{u}^{(n)} \\
&= \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\cdot\mathbf{L}_{\mathbf{T}}\cdot\mathbf{X}_{(n)}^T\mathbf{u}^{(n)}.
\end{aligned} \tag{5.18}$$

Therefore, the between-class scatter matrix of tensor samples, denoted $\mathbf{S}_b$, can be represented as:

$$\mathbf{S}_b = \mathbf{S}_t - \mathbf{S}_c = \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left(\mathbf{L_T} - \mathbf{L_W} \right) \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}. \tag{5.19}$$

Under the assumption that $\left\{ \mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(n-1)}, \mathbf{u}^{(n+1)}, \cdots, \ \mathbf{u}^{(N)} \right\}$ are know, we can encode discriminant information by maximizing the objective function as follows:

$$\begin{aligned} &\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left[\left(\mathbf{L_T} - \mathbf{L_W} \right) - \lambda \mathbf{L_W} \right] \mathbf{X}_{(n)}^T \mathbf{u}^{(n)} \\ \Leftrightarrow &\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left(\mathbf{L_T} - \lambda \mathbf{L_W} \right) \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}, \lambda \geq 1, \end{aligned} \tag{5.20}$$

where λ balances the importance of the between- and within-class variances.

5.2.3 Orthogonal Multilinear Transformation

To preserve the similarity and diversity of neighboring samples and encode discriminant information, we combine objective functions in Eqs. (5.14) and (5.20) and derive the objective function of the TLDP-MMC as follows:

$$\begin{aligned} &\min_{\mathbf{u}^{(n)}} \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left[\mathbf{L_C} - \left(\mathbf{L_T} - \lambda \mathbf{L_W} \right) \right] \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}, \\ &\text{s.t. } \mathbf{y} \mathbf{D_S} \mathbf{y}^T = \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T \mathbf{u}^{(n)} = 1. \end{aligned} \tag{5.21}$$

To obtain the optimal solution $\mathbf{u}^{(n)}$ to Eq. (5.21), we solve the generalized eigenvalue problem as follows:

$$\mathbf{X}_{(n)} \left[\mathbf{L_C} - \left(\mathbf{L_T} - \lambda \mathbf{L_W} \right) \right] \mathbf{X}_{(n)}^T \mathbf{u}^{(n)} = \lambda \mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}. \tag{5.22}$$

The eigenvectors of Eq. (5.22) corresponding to the P_n smallest eigenvalues, as the basis vectors of the P_n-dimensional subspace, constitute transformation matrix $\mathbf{U}^{(n)}$ along the n-th mode. However, basis vectors in $\mathbf{U}^{(n)}$ are not orthogonal since matrix $\left(\mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T \right)^{-1}$ is not symmetry. As introduced in [129], the orthogonality of basis vectors can enhance

both locality-preserving and discriminant power. Therefore, to ensure $\left(\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, \cdots, \mathbf{u}_{P_n}^{(n)}, \right)$ are normal, we involve an orthonormal constraint on the objective function in Eq. (5.21) as follows:

$$\mathbf{u}_i^{(n)^T} \mathbf{u}_j^{(n)} = \begin{cases} 0, \text{ if } i \neq j, \\ 1, \text{ if } i = j. \end{cases} \tag{5.23}$$

We utilize an iterative way to acquire basis vectors by assuring that new transformation vector $\mathbf{u}_k^{(n)}$, $k \leq P_N$, is orthonormal to all previously obtained vectors $\left\{\mathbf{u}_1^{(n)}, \cdots, \mathbf{u}_{k-1}^{(n)}\right\}$ in $\mathbf{U}^{(n)}$. The first transformation vector, $\mathbf{u}_1^{(n)}$, calculated from Eq. (5.22), is the eigenvector of $\mathbf{M}_1$ that corresponds to the smallest eigenvalue. The representation of $\mathbf{M}_1$ is shown as follows:

$$\mathbf{M}_1 = \left(\mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T\right)^{-1} \cdot \left\{\mathbf{X}_{(n)} \left[\mathbf{L_C} - (\mathbf{L_T} - \lambda \mathbf{L_W})\right] \mathbf{X}_{(n)}^T\right\}. \tag{5.24}$$

Based on $\mathbf{u}_1^{(n)}$, basis vector $\mathbf{u}_k^{(n)}$, $k > 1$, can be iteratively calculated as the eigenvector of $\mathbf{M}_k$ associated with the smallest eigenvalue. The mathematical expression of $\mathbf{M}_k$ is shown as follows:

$$\mathbf{M}_k = \left[\mathbf{I}_{I_n} - \left(\mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T\right)^{-1} \cdot \mathbf{U}_{k-1}^{(n)} \left(\mathbf{V}_{k-1}^{(n)}\right)^{-1} \left(\mathbf{U}_{k-1}^{(n)}\right)^T\right] \mathbf{M}_1 \tag{5.25}$$

where matrices $\mathbf{U}_{k-1}$ and $\mathbf{V}_{k-1}$ are defined as:

$$\begin{aligned} \mathbf{U}_{k-1}^{(n)} &= \left[\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, \cdots, \mathbf{u}_{k-1}^{(n)}\right], \\ \mathbf{V}_{k-1}^{(n)} &= \mathbf{U}_{k-1}^{(n)^T} \left(\mathbf{X}_{(n)} \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_M} \mathbf{D_S} \mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_M} \mathbf{X}_{(n)}^T\right)^{-1} \mathbf{U}_{k-1}^{(n)}. \end{aligned} \tag{5.26}$$

On the basis of Eqs. (5.24) and (5.25), we can obtain orthogonal projection matrix $\mathbf{U}^{(n)}$ along the n-th mode. In an iterative manner, we can obtain the multilinear transformation under the criteria of TOLDP-MMC. The details are shown in Algorithm 4.

Algorithm 4 Pseudo-code of the TOLDP-MMC

Input: a set of tensor samples $\{\boldsymbol{\mathcal{X}}_{1,1}, \cdots, \boldsymbol{\mathcal{X}}_{1,N_1} | \boldsymbol{\mathcal{X}}_{2,1}, \ \cdots, \boldsymbol{\mathcal{X}}_{2,N_2} | \cdots | \boldsymbol{\mathcal{X}}_{C,1}, \cdots, \boldsymbol{\mathcal{X}}_{C,N_1}\} \subset \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, belong to C classes

Output: Lower dimensional features of input tensor samples, $\left\{\boldsymbol{\mathcal{Y}}_{c,i} \in \mathbb{R}^{P_1 \times P_2 \times \cdots \times P_N}, c = 1, 2, \cdots, C, i = 1, 2, \cdots, N_c\right\}$, extracted by transformation matrices $\left\{\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times P_n}, \ P_n < R_n, n = 1, 2, \cdots, N\right\}$

Initialization:

1: **for** $n \leftarrow 1$ to N **do**

2: Initialize projection matrix $\mathbf{U}^{(n)}$ using the eigenvectors of matrix $\mathbf{X}_{(n)}\mathbf{X}_{(n)}^T$ associated with the largest P_n eigenvalues

3: **end for**

4: Calculate mapped samples $\boldsymbol{\mathcal{Y}}_{c,i} = \boldsymbol{\mathcal{X}}_{c,i} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}$ and $\Psi_0 = \sum_{c=1}^{C} \sum_{i=1}^{N_c} \|\boldsymbol{\mathcal{Y}}_{c,i}\|_F^2$

5:

Optimization:

6: **for** $t \leftarrow 1$ to T **do**

7: **for** $n \leftarrow 1$ to N **do**

8: With $\left\{\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(n-1)}, \mathbf{U}^{(n+1)}, \cdots, \mathbf{U}^{(N)}\right\}$ are fixed, iteratively construct $\widetilde{\mathbf{U}}^{(n)}$ using the orthonormal eigenvectors of Equations (5.24) and (5.25) associated with the smallest eigenvalues

9: $\mathbf{U}^{(n)} \leftarrow \widetilde{\mathbf{U}}^{(n)}$

10: **end for**

11: Calculate mapped samples and Ψ_t

12: **if** $(\Psi_t - \Psi_{t-1}) \leq \zeta$? **then** Break

13: **end if**

14: **end for**

5.3 TOLDP-MMC in Gait Recognition and Salt-dome Tracking

5.3.1 Gait Recognition

To evaluate the performance of the TOLDP-MMC on feature extraction, we apply this method on gait recognition. In the experiments, we use a subset of the USF HumanID "gait challenge" data sets [130], which contain the binary tensor samples of gait silhouette from 71 subjects walking along elliptical paths. Each tensor sample has a size of $128 \times 88 \times 10$, and one tensor example is illustrated in Figure 5.3. The gait silhouettes of each subject are captured under various conditions, which are decided by three variants: the viewpoint (left or right), shoes (type A or B), and the surface (grass or concrete). In gait recognition, we

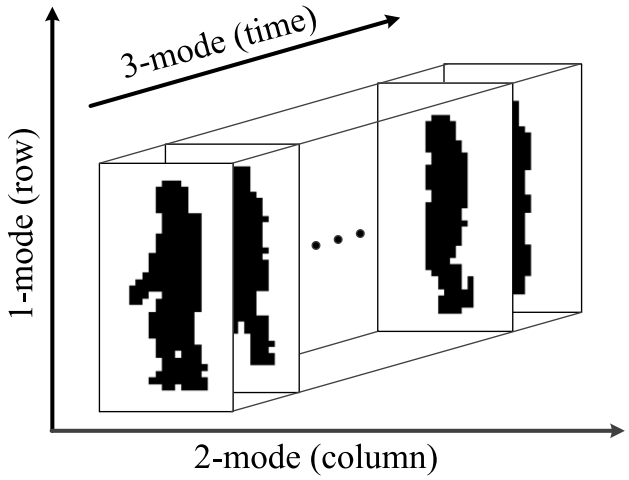


Figure 5.3: A binary tensor sample contains gait silhouette.

Table 5.1: The capturing conditions of training and testing data sets.

Data Sets	Training Set	Testing Sets						
		A	B	C	D	E	F	G
Capturing Conditions	GAR	GAL	GBR	GBL	CAR	CBR	CAL	CBL
Subject Number	71	71	41	41	70	44	70	44

first obtain projection matrices along all modes by applying tensor-based subspace learning methods to a training data set. To determine the labels of samples in testing data sets, we compare the features of samples in testing data sets with those in the training data set. Different data sets are captured under different conditions, and subjects in all data sets are unique. Therefore, every tensor sample is unique. We list the number of subjects and the capturing conditions of training and testing data sets in Table 5.1, where C, G, A, B, L, and R stand for the concrete surface, the grass surface, shoes with type A, shoes with type B, the left view, and the right view, respectively. In the training data set, the gait silhouettes of all 71 subjects are captured under the condition of GAR. In contrast, the number of subjects In testing data sets may be less than 71.

In the experiments, we compare the performance of the TOLDP-MMC with typical tensor-based subspace learning methods, such as MPCA [130], TOLPP [131], and TMMC [131]. To evaluate their performance on gait recognition, we use two metrics, rank-1 and rank-5. Rank-1 and rank-5 represent the frequency of training samples that truly match testing sam-

Table 5.2: Compare the performance of different subspace learning methods on gait recognition.

Metrics	Methods	Dimensions	A	B	C	D	E	F	G	Average
Rank-1(%)	TPCA	90% Energy	61.76	**51.30**	37.62	**21.75**	17.92	**14.20**	**14.68**	**31.32**
	TMMC	[20, 13, 5]	60.52	46.34	32.38	12.83	12.59	12.32	10.70	27.86
	TOLPP	[22, 14, 5]	60.39	**51.30**	**38.81**	20.97	15.74	14.04	10.70	30.28
	TOLDP-MMC	[22, 14, 6]	**62.04**	51.06	36.67	21.13	**19.61**	13.10	12.44	30.86
Rank-5(%)	TPCA	90% Energy	41.98	34.33	**24.48**	**14.77**	14.38	10.83	11.34	21.73
	TOMMC	[22, 15, 5]	**47.15**	**36.36**	24.43	12.27	10.46	9.55	8.86	21.30
	TOLPP	[22, 14, 5]	44.02	34.56	24.05	14.52	14.38	11.64	11.24	22.06
	TOLDP-MMC	[21, 14, 6]	45.06	34.75	24.19	14.71	**14.82**	**11.89**	**11.39**	**22.40**

ples in the top one and five matches. In Table 5.2, we list the all metric values of different methods and the corresponding dimensions when these values achieve the maximum. For TPCA, we list the energy kept instead of the optimal dimensions. We notice that the rank-1 value of TPCA is higher than the TOLDP-MMC. However, in contrast, the rank-5 value of the TOLDP-MMC is better, which shows the robustness of the TOLDP-MMC.

5.3.2 Salt-dome Tracking

In this section, we utilize the TOLDP-MMC to implement the salt-dome tracking. The block diagram of the salt-dome tracking method based on the TOLDP-MMC is shown in Figure 5.4.

Extracting Texture Features

On the basis of salt-dome boundaries labeled by interpreters in reference sections, we attempt to obtain texture features that involve spatial information on all directions. In the tracking method using the TOLDP-MMC, we build texture tensors from the labeled boundaries of reference sections. We define N_r neighboring seismic sections as reference sections, in which the corresponding boundaries, denoted $\mathbf{l}_b$, $b = 1, 2, \cdots, N_r$, are manually labeled. Points on these reference boundaries have the coordinate vectors of $\mathbf{l}_{b,k}$, $k = 1, 2, \cdots, K_b$, where K_b is the length of $\mathbf{l}_b$. To fully capture texture information along all reference boundaries, we focus on texture patches centered at boundary points with a size of $N_p \times N_p$. Therefore, third-order texture tensors can be built from these patches.

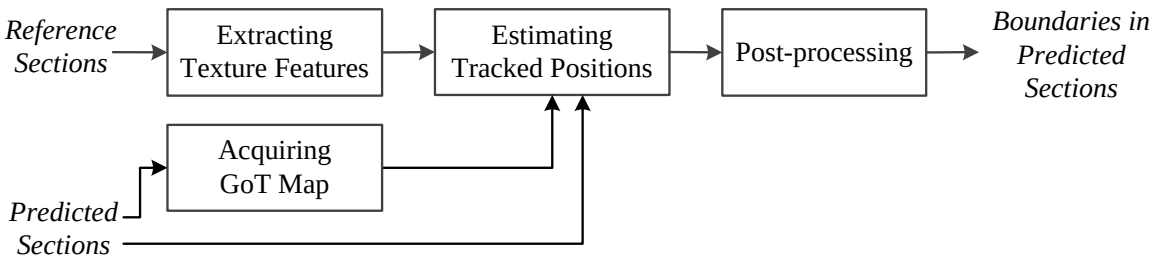


Figure 5.4: The block diagram of the tracking method based on the OLDP-MMC.

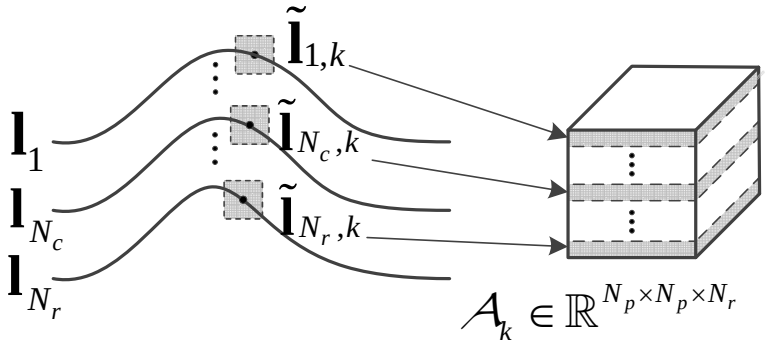


Figure 5.5: The texture tensor built from the boundaries of reference sections

To ensure the equality on both tracking directions, we first define the centers of texture tensors as points on the centric reference boundary $\mathbf{l}_{N_c}$, where $N_c = \lceil N_r/2 \rceil$. Then, to construct texture tensors, we need to identify the corresponding patch centers in neighboring reference sections. The localization of patch centers on $\mathbf{l}_b$, $b \neq N_c$, is shown as follows:

$$\tilde{\mathbf{l}}_{b,k} = \arg\min_{\mathbf{l}_{b,t}} \left\| \mathbf{l}_{b,t} - \mathbf{l}_{N_c,k} \right\|_2, t = 1, 2, \cdots, K_b. \tag{5.27}$$

Based on the definitions of $\mathbf{l}_{N_c}$ and Eq. (5.27), $\tilde{\mathbf{l}}_{N_c,k}$ is equal to $\mathbf{l}_{N_c,k}$. Therefore, boundary points on the centric section identify groups of patch centers, denoted $\left\{ \tilde{\mathbf{l}}_{b,k}, b = 1, 2, \cdots, N_r \right\}$, $k = 1, 2, \cdots, K_{N_c}$. By stacking texture patches belonging to the same group along the third direction as Figure 5.5 shows, we can construct third-order texture tensors from reference sections, which are denoted as $\left\{ \boldsymbol{\mathcal{A}}_k \in \mathbb{R}^{N_p \times N_p \times N_r}, k = 1, 2, \cdots, K_{N_c} \right\}$.

Although current texture tensors contain texture information from all reference sections, only one tensor may not be enough to reflect the changes of textures along reference boundaries. Therefore, to utilize texture information along local boundaries, we introduce a

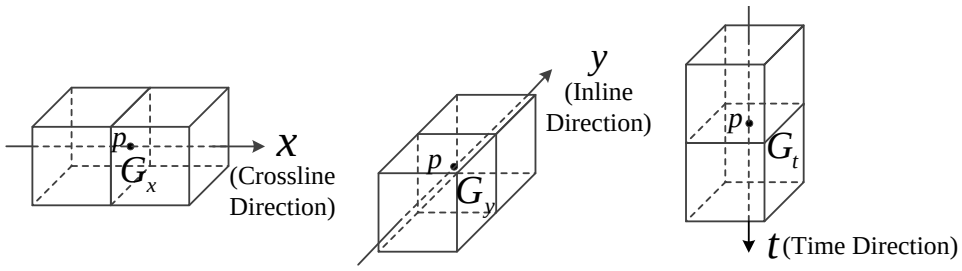


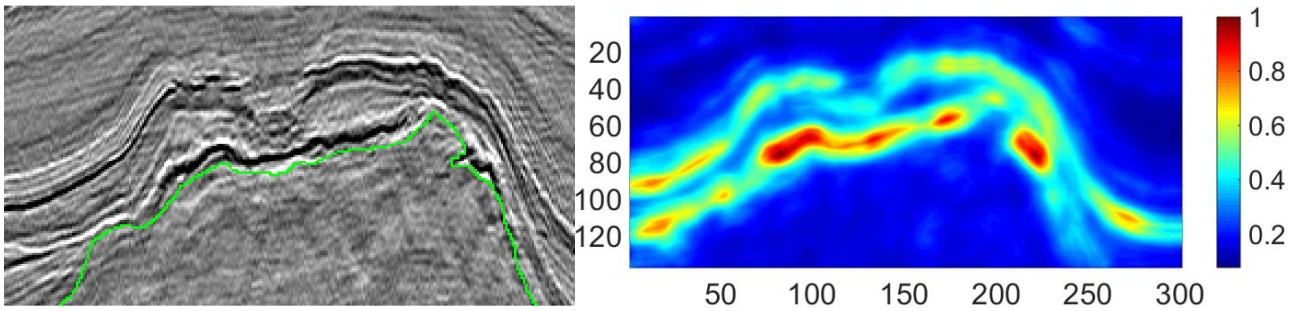
Figure 5.6: Neighboring cubes defined around p in three directions

tensor group centered at tensor $\boldsymbol{\mathcal{A}}_k$, denoted $\boldsymbol{\mathcal{G}}_k = \{\boldsymbol{\mathcal{A}}_{k-N_s}, \cdots, \boldsymbol{\mathcal{A}}_k, \cdots, \boldsymbol{\mathcal{A}}_{k+N_s}\}$, where N_s determines the window size along boundaries. Therefore, all tensors extracted from reference boundaries can be classified into tensor groups according to their proximity. By applying the OLDP-MMC on classifed tensor groups, we can obtain projection matrices on each mode, denoted $\mathbf{U}_k^{(1)} \in \mathbb{R}^{J_{1,k} \times N_p}$, $\mathbf{U}_k^{(2)} \in \mathbb{R}^{J_{2,k} \times N_p}$, and $\mathbf{U}_k^{(3)} \in \mathbb{R}^{J_{3,k} \times N_r}$, where $J_{n,k}$, $n = 1, 2, 3$, are the dimensions of projected column subspace. By projecting $\boldsymbol{\mathcal{G}}_k$ onto these matrices, we can obtain texture features in the form of tensors, denoted $\tilde{\boldsymbol{\mathcal{G}}}_k$, which has the element calculated as follows:

$$\tilde{\boldsymbol{\mathcal{A}}}_m = \boldsymbol{\mathcal{A}}_m \times_1 \mathbf{U}_k^{(1)} \times_2 \mathbf{U}_k^{(2)} \times_3 \mathbf{U}_k^{(3)}, m \in \{k - N_s, \cdots, k + N_s\}\,. \tag{5.28}$$

Acquiring GoT Maps

Using texture features extracted from reference boundaries, we can estimate the position of boundary points in predicted sections. However, migrated seismic data commonly involve noise, which may effect the accuracy of tracked boundary. Therefore, to increase the robustness of the tracking method, we select the 3D multi-scale GoT attribute [109] as a constraint because of its capability of describing the texture difference between the salt dome and its neighboring rock strata. For each point p in the predicted section, we define three pairs of cubes surrounding it as Figure 5.6 shows. The texture difference between neighboring cubes represents the texture gradient along one direction, denoted G_i, $i \in \{x, y, t\}$. By combining these texture gradients, we can obtain the GoT value, which is



(a) Reference image (b) GoT map

Figure 5.7: The reference image (inline 399) and its corresponding GoT map

calculated as $G = \sqrt{G_x^2 + G_y^2 + G_t^2}$. According to the work of [109], texture gradients are consistent with the perception of interpreters and can be represented as follows:

$$G_i = E\left(\left|\mathscr{F}\left\{\left|\mathscr{F}\left\{\text{abs}\left(\mathbf{W}_{i-} - \mathbf{W}_{i+}\right)\right\}\right|\right\}\right), i \in \{x, y, t\}, \tag{5.29}$$

where $|\mathscr{F}\{\cdot\}|$ represents the magnitude of 3D Fourier transform, $\{\mathbf{W}_{i-}, \mathbf{W}_{i+}\}$ defines the pair of neighboring cubes along one direction, and mean operator $E(\cdot)$ pools the difference cube into a single value. Figure 5.7 shows one seismic section with the manually labeled salt-dome boundary in green and its corresponding GoT map. We notice that the blue area at the bottom of the GoT map roughly illustrates the salt body. In contrast, yellow and red regions indicate surrounding rock strata.

Estimating Tracked Positions

Using the GoT map and texture features extracted from reference boundaries, we can estimate the initial positions of tracked points. To ensure the computational efficiency of the tracking process, we project the labeled boundary of only the centric reference section onto the predicted section and keep the coordinates of all boundary points unchanged. To identify the optimal tracked position, we search along the normal direction of the projected point within a radius of $(2R_s + 1)$, where R_s is determined by the distance between the predicted section and the centric reference section. However, if the shape of the salt dome

drastically changes among neighboring seismic sections, we may not be able to have access to the boundary area by searching around the current predicted point. Therefore, we need to shift projected points under the constraint of the GoT map. The shifting strategy with two thresholds is shown as follows:

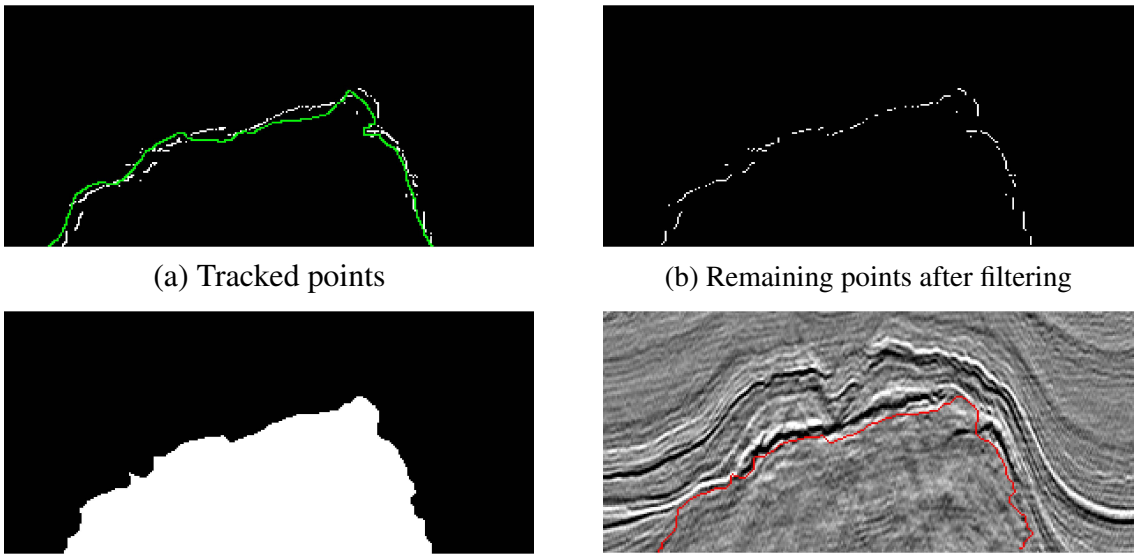
$$\hat{\mathbf{l}}_{N_c,k} = \begin{cases} \tilde{\mathbf{l}}_{N_c,k} - R_s \mathbf{v}_\perp, & \bar{\mathbf{G}}\left(\tilde{\mathbf{l}}_{N_c,k}\right) > T_H \\ \tilde{\mathbf{l}}_{N_c,k} + R_s \mathbf{v}_\perp, & \bar{\mathbf{G}}\left(\tilde{\mathbf{l}}_{N_c,k}\right) < T_L \\ \tilde{\mathbf{l}}_{N_c,k}, & \text{Otherwise} \end{cases}, \tag{5.30}$$

where unit vector $\mathbf{v}_\perp$ indicates the normal direction of point $\tilde{\mathbf{l}}_{N_c,k}$ and $\bar{\mathbf{G}}\left(\tilde{\mathbf{l}}_{N_c,k}\right)$ represents the averaged GoT value of the 3×3 neighborhood of point $\tilde{\mathbf{l}}_{N_c,k}$. T_H and T_L defines two thresholds. If the averaged GoT value of $\tilde{\mathbf{l}}_{N_c,k}$ is greater than T_H, it means that the initially projected point is located in surrounding rock strata and needs to be shifted towards the salt body. In contrast, if the averaged GoT value of $\tilde{\mathbf{l}}_{N_c,k}$ is less than T_L, it means that the initially projected belongs to the salt body and needs to be shifted towards the boundary area.

On the basis of projected point $\hat{\mathbf{l}}_{N_c,k}$, we define a group of potential tracked points, denoted $\hat{\mathbf{l}}^{(s)}_{N_c,k}$, $s = 1, 2, \cdots, (2R_s + 1)$. For each potential tracked point, we randomly select N_r points from its $\lceil\sqrt{N_r}\rceil \times \lceil\sqrt{N_r}\rceil$ neighborhood, which represent the centers of texture patches with a size of $N_p \times N_p$. By stacking these patches along the third direction, we can obtain texture tensor $\boldsymbol{\mathcal{P}}^{(s)}_{N_c,k}$. As Eq. (5.28) shows, we extract the texture features of $\boldsymbol{\mathcal{P}}^{(s)}_{N_c,k}$, denoted $\tilde{\boldsymbol{\mathcal{P}}}^{(s)}_{N_c,k} \in \mathbb{R}^{J_{1,k} \times J_{2,k} \times J_{3,k}}$, using projection matrices $\left\{\mathbf{U}^{(1)}_k, \mathbf{U}^{(2)}_k, \mathbf{U}^{(3)}_k\right\}$. The difference between $\tilde{\boldsymbol{\mathcal{P}}}^{(s)}_{N_c,k}$ and $\tilde{\boldsymbol{\mathcal{G}}}_k$ can be calculated as follows:

$$d^{(s)}_{N_c,k} = \left\| \tilde{\boldsymbol{\mathcal{G}}}_k - \tilde{\boldsymbol{\mathcal{P}}}^{(s)}_{N_c,k} \right\|_F = \left(\sum \left\| \tilde{\boldsymbol{\mathcal{A}}}_m - \tilde{\boldsymbol{\mathcal{P}}}^{(s)}_{N_c,k} \right\|^2_F \right)^{1/2}. \tag{5.31}$$

The potential position with the smallest difference is the tracked position. Fig 5.8(a) illustrates the tracked points of the predicted section inline 389, which are estimated from the



(a) Tracked points

(b) Remaining points after filtering

(c) Highlighted salt body

(d) Tracked boundary

Figure 5.8: The post-processing steps to synthesize the tracked boundary

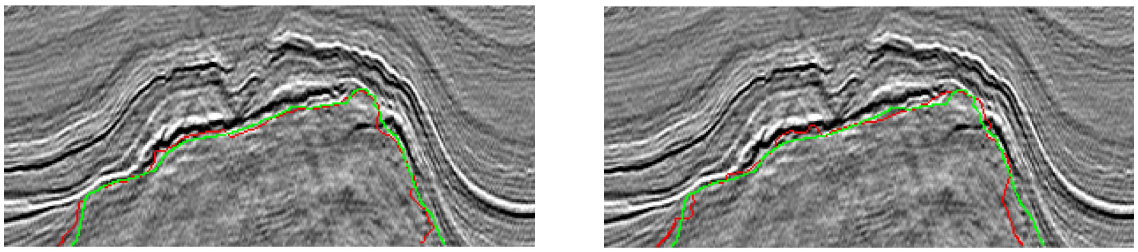
green labeled boundary of inline 399.

Post-processing

On the basis of tracked points, to synthesize the tracked boundary, we need to apply necessary post-processing steps. We first use the median filter to remove noisy points. By connection these remaining points shown in Figure 5.8(d), we can highlight the salt body. To prevent from synthesizing the jagged boundary, we apply the closing operation on the highlighted salt body with a disk structuring element as Figure 5.8(c) illustrates. The boundary extracted from Figure 5.8(c) is shown in Figure 5.8(d) as the tracked boundary in red.

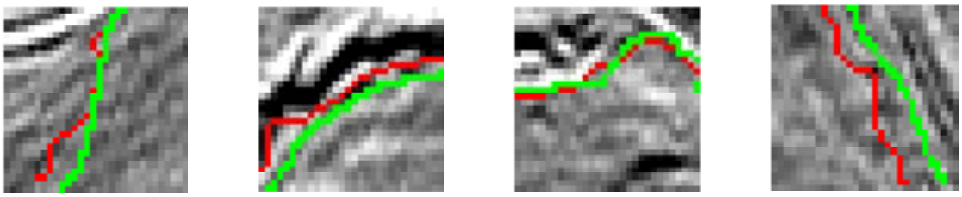
Experimental Results

In this section, we apply the salt-dome tracking method on a 3D real seismic dataset acquired from the Netherlands offshore F3 block with the size of 24×16 km^2 in the North Sea [116]. To illustrate the performance of the tracking method based on the TOLDP-MMC, we focus on a local volume of the dataset containing discernible salt-dome structures. The tested volume has an inline number ranging from 389 to 409, a crossline number

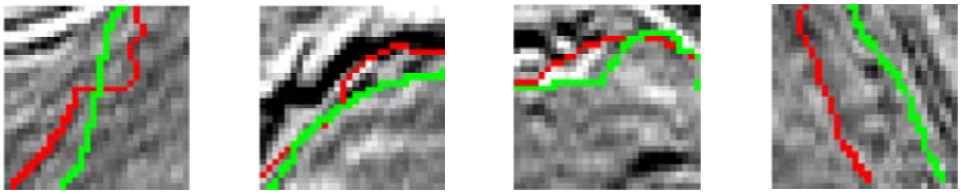


(a) (b)

Figure 5.9: The comparison between the green ground truth and the red tracked boundaries of inline 389 synthesized by (a) the tracking method based on the TOLDP-MMC and (b) the tracking method in Chapter 4



(a) Details of the tracked boundary in Figure 5.9(a)



(b) Details of the tracked boundary in Figure 5.9(b)

Figure 5.10: The details of the green ground truth and the red tracked boundaries in Figure 5.9

ranging from 401 to 701, and a time direction ranging from 1,300ms to 1,848ms with a step of 4ms. Figures. 5.7(a) and 5.8(d) illustrate seismic sections extracted from the local volume.

As we mentioned in previous sections, we first select seven reference sections with the inline number ranging from 396 to 402, the salt-dome boundaries in which are labeled by interpreters. Then we build texture tensors with a size of $20 \times 20 \times 7$ along labeled boundaries and extract texture features on the basis of Eq. (5.28). In the calculation of the GoT map, we define the size of cubes as $7 \times 7 \times 7$. In the tracking process, we empirically define T_H and T_L as 0.6 and 0.2, respectively. Search radius R_s is proportional

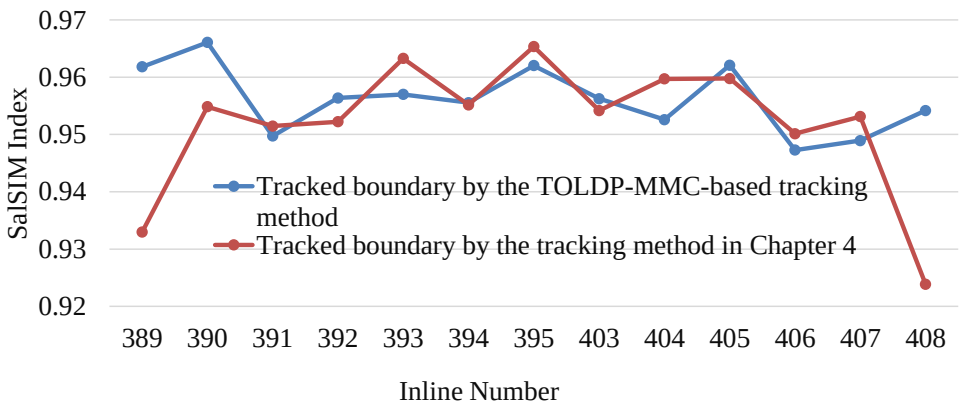


Figure 5.11: The SalSIM indices of tracked boundaries ranging from 389 to 395 and 403 to 409

to the offset between the predicted and the centric reference sections. In Figure 5.9, we compare the green manually labeled ground truth with red tracked boundaries in inline 389 synthesized by the salt-dome tracking method based on the TOLDP-MMC and one introduced in Chapter 4, respectively. Figure 5.10 shows the more details of the comparison in Figure 5.9. We notice that the tracked boundary synthesized by the tracking method based on the TOLDP-MMC is more similar to the ground truth, in contrast to the one obtained in Chapter 4, especially around the left- and right-bottom. To objectively evaluate the similarity between tracked boundaries and the ground truth, we use the SalSIM index introduced in Chapter 4, which is derived from the Frèchet distance. In our experiment, we synthesize tracked boundaries in inline 389 to 395 and 403 to 409 using the tracking method based on the TOLDP-MMC and the method in Chapter 4. We noticed that the salSIM index of boundaries synthesized by the memthod in Chapter 4 has a decreasing trend with the

Table 5.3: The objective comparison between the salt-dome tracking method based on the TOLDP-MMC and the tracking method in Chapter 4

Tracking Method	Mean	Standard Deviation	AMD (pixel)	Elapsed CPU Time (s)
Tracking Method based on the TOLDP-MMC	0.9559	0.0054	8.57	14.71
Tracking Method in Chapter 4	0.9508	0.0115	9.32	70.82

increasing offset to the centric reference section. However, salt boundaries obtained by the tracking method based on the TOLDP-MMC yield more stable salSIM indices. Table 5.3 shows the statistics of the salSIM indices in Figure 5.11, in which the averaged maximum distance (AMD) represents the mean of the Frèchet distance of the tracked boundaries to the ground truth. We implement both methods on a computer with Core i7-3720QM CPU at 2.60GHz and 12GB RAM and list the corresponding run-time in Table 5.3. We noticed that the tracking method based on the TOLDP-MMC has higher mean value, lower standard deviation, AMD, and computational complexity. In addition, the tracking method based on the TOLDP-MMC also has higher potential of being implemented in parallel, which can further increases interpretation efficiency.

5.4 Summary

In this chapter, we introduced a tensor-based subspace learning method, TOLDP-MMC, to extract features from multidimensional data. Different from the TLPP, this algorithm consists of three parts, the tensor locality-preserved diversity projection, the maximum margin criterion, and orthogonality. The locality-preserved diversity projection maintains the relative position information of neighboring samples, while embedding neighboring samples in the subspace as closely as possible. The maximum margin criterion encodes global discriminant information and separates samples from different classes as far as possible by maximizing the difference between the within- and between-class variances of embedded samples. To increase the discriminating power, we orthogonalized the bases of embedding spaces. We applied the TOLDP-MMC on gait recognition and salt-dome tracking. Experimental results showed the discriminating power of the TOLDP-MMC improved performance and accuracy.

CHAPTER 6
CONCLUSION

In Chapter 3, we introduced an interactive interpretation framework that delineated faults in seismic volumes using their geological and geometric features. The geological feature of faults is their discontinuity along horizons. The geometric feature of faults is their line-like or curved shapes in seismic sections, which appear as curved surfaces in seismic volumes. To improve the efficiency of fault interpretation, we classified seismic sections into reference and predicted ones by borrowing the concept of I- and B-frames in video-coding techniques and applied different strategies on the two types of sections. In the reference sections, we utilized discontinuity maps to highlight discontinuous regions along horizons, referred to as likely fault regions, and employed the Hough transform to extract the line features of faults. Because of the appearance of noise and the limitation of the Hough transform, it was inevitable that detected results contain some false features that violate geological constraints. To remove false features, we characterized geological constraints using the spatial relationships of detected features. The connection of remaining features determines the accuracy of fault labeling. Simply connecting features using lines depicts a rough shape of a fault. However, such labeling is not accurate since lines connecting features involve no geological information. To improve interpretation accuracy, we searched horizontally along the initially labeled fault for the highest discontinuity value and obtained another candidate. The combination of these two candidates forms the optimal labelling of faults in reference sections. For predicted sections, we synthesized tracked faults using ones detected from two reference sections for high accuracy. We utilized the discontinuity attribute to estimate tracking vectors, which projected the local segments of faults labeled in reference sections to the corresponding position in the target predicted section. The optimal combination of fault segments projected from reference sections involves discontinuity

and spatial constraints. Faults labeled in reference and predicted sections form the fault surface in the seismic volume. To evaluate the performance of the interactive framework on fault interpretation, we introduced the fault similarity (FauSIM) index that describes the similarity between detected faults and manually picked faults. The FauSIM index based on the Fréchet distance compares both the local and global structures of faults. Experimental results showed that the interactive framework has the capability to accurately detect faults in seismic sections, and the objective FauSIM indices complied with the subjective observation of interpreters.

In Chapter 4, we extended the concept of the interactive interpretation framework introduced in Chapter 3 to the labeling of salt domes. In a seismic volume, we classified its seismic sections containing salt domes into reference and predicted ones, and applied different strategies for these two types of sections. Since salt domes containing chaotic reflections have different texture from surrounding structures formed by other types of rocks, to capture the change of texture between salt and non-salt regions, we calculated the GoT attribute of all sections by involving a perceptual dissimilarity measure. In reference sections, we highlighted salt-dome boundaries with higher GoT values using a threshold automatically generated from Ostu's method. From an interactively selected initialization point inside the salt dome, we performed region growing, and regions indicating the salt dome keep growing and stop at boundary regions. After the closing operation, grown regions determine the boundaries of the salt dome in reference sections. By tracking detected boundaries in reference sections that constitute the minority of seismic sections through a seismic volume, we further improve the efficiency of salt-dome interpretation. Since local areas along salt-dome boundaries commonly have similar textures, we built texture tensors by stacking texture patches around salt-dome boundaries in the reference section along the third mode. While building tensors, we introduced the incremental tensor-based PCA to implement tensor classification. Although seismic data commonly contain noise, noise-adjusted tensor-based PCA generated a set of transform matrices by selecting PCs with

the highest SNR. To localized boundary points in predicted sections, we first projected the detected boundary in the reference section onto the target predicted section and kept the coordinates of all points unchanged. Then, to identify the tracked position of every projected point, we searched among candidate points located along the normal direction of the projected boundary. Every candidate point corresponded to a patch pair, which could be reconstructed by obtained transform matrices. The candidate point with the smallest reconstruction error is selected as the tracked point. To evaluate the performance of the interactive framework on salt-dome interpretation, we introduced the salt-dome similarity (SalSIM) index that described the similarity between detected and manually picked salt-dome boundaries. The SalSIM index based on the Fréchet distance compares both the local and global structures of salt-dome boundaries. Experimental results showed that the interactive framework has the capability to accurately detect salt-domes in the seismic volume, and the objective SalSIM indices complied with the subjective observation of interpreters.

In Chapter 5, we introduced a tensor-based subspace learning method, TOLDP-MMC, to extract features from multidimensional data. Different from the TLPP, this algorithm consists of three parts, the tensor locality-preserved diversity projection, the maximum margin criterion, and orthogonality. The locality-preserved diversity projection maintains the relative position information of neighboring samples, while embedding neighboring samples in the subspace as closely as possible. The maximum margin criterion encodes global discriminant information and separates samples from different classes as far as possible by maximizing the difference between the within- and between-class variances of embedded samples. To increase the discriminating power, we orthogonalized the bases of embedding spaces. We applied the TOLDP-MMC on gait recognition and salt-dome tracking. Experimental results showed the discriminating power of the TOLDP-MMC improved performance and accuracy.

Appendices

APPENDIX A

DERIVATIONS OF TOTAL AND WITHIN-CLASS SCATTER MATRICES OF TENSOR SAMPLES.

In supervised learning, we assume that M tensor samples belong to C classes and modify the notation of samples as $\boldsymbol{\mathcal{X}}_{c,i}$, where $c = 1, 2, \cdots, C$, indicates the class label and $i = 1, 2, \cdots, N_c$, represents the sample index in class c. The n-th mode unfolding matrix of $\boldsymbol{\mathcal{X}}_{c,i}$ is $\mathbf{X}_{c,i(n)}$. $\mathbf{X}_{c(n)} = \left[\mathbf{X}_{c,1(n)}, \mathbf{X}_{c,2(n)}, \cdots, \mathbf{X}_{c,N_c(n)}\right]$ concatenates all n-th mode unfolding matrices in class c, and $\overline{\mathbf{X}}_{c(n)}$ is the mean of all n-th mode unfolding matrices in class c. $y_i = \boldsymbol{\mathcal{X}}_i \times_1 \mathbf{u}^{(1)^T} \times_2 \mathbf{u}^{(2)^T} \cdots \times_N \mathbf{u}^{(N)^T}$, $i = 1, 2, \cdots, M$, is mapped from sample $\boldsymbol{\mathcal{X}}_i$, where $\left\{\mathbf{u}^{(n)} \in \mathbb{R}^{I_n \times 1}, n = 1, 2, \cdots, N\right\}$ are a set of transformation vectors for N modes. Here, y_i still represents a tensor, although each mode has only one dimension. The mean of y_i, $i = 1, 2, \cdots, N_c$, in class c is denoted as $\bar{y}_c$. Similar to the matrix representation of the HOSVD in Eq. (4.15), we denote y_i as follows:

$$\begin{aligned} y_i &= \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \mathbf{u}_{\boldsymbol{\Phi}^{(n)}} \\ &= \mathbf{u}^{(n)^T} \mathbf{X}_{i(n)} \left(\mathbf{u}^{(n+1)} \otimes \cdots \otimes \mathbf{u}^{(N)} \otimes \mathbf{u}^{(1)} \otimes \cdots \otimes \mathbf{u}^{(n-1)}\right), \end{aligned} \tag{A.1}$$

For simplicity, we define $\mathbf{U}_{\boldsymbol{\Phi}_{(n)}}^{\mathbf{I}_M} = \mathbf{I}_M \otimes \mathbf{u}_{\boldsymbol{\Phi}_{(n)}}$, where matrix $\mathbf{I}_M$ represents an identity matrix with a size of M. The transpose of matrix $\mathbf{U}_{\boldsymbol{\Phi}_{(n)}}^{\mathbf{I}_M}$ is denoted as $\mathbf{U}_{\boldsymbol{\Phi}_{(n)}^T}^{\mathbf{I}_M}$. In addition, when deriving total- and within-class variance of tensor samples, we need to define three constant items, identity matrices $\mathbf{I}_{N_c}$ and $\mathbf{I}_{\Pi(n)}$ with the sizes of N_c and $\prod_{k \neq n} I_k$, respectively, and vector $\mathbf{e}_{N_c}$ with all N_c entries equal to one. The following sections will introduce the derivation process in detail.

A.1 Within-class Scatter Matrix of Tensor Samples

The within-class scatter matrix of tensor samples in Eq. (5.16) can be simplified as follows:

$$
\begin{aligned}
\mathbf{S}_w &= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\left(y_i-\bar{y}_c\right)\left(y_i-\bar{y}_c\right)^T \\
&= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\mathbf{u}^{(n)^T}\left(\mathbf{X}_{c,i(n)}-\overline{\mathbf{X}}_{c(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{X}_{c,i(n)}^T-\overline{\mathbf{X}}_{c(n)}^T\right)\mathbf{u}^{(n)} \\
&= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\mathbf{u}^{(n)^T}\left[\mathbf{X}_{c,i(n)}-\mathbf{X}_{c(n)}\left(\frac{1}{N_c}\mathbf{e}_{N_c}\otimes\mathbf{I}_{\Pi(n)}\right)\right]\mathbf{u}_{\mathbf{\Phi}^{(n)}}\cdot \\
&\qquad\qquad \mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left[\mathbf{X}_{c,i(n)}^T-\left(\frac{1}{N_c}\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{X}_{c(n)}^T\right]\mathbf{u}^{(n)} \\
&= \sum_{c=1}^{C}\sum_{i=1}^{N_c}\mathbf{u}^{(n)^T}\Big[\mathbf{X}_{c,i(n)}\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\mathbf{X}_{c,i(n)}^T \\
&\quad -\frac{1}{N_c}\mathbf{X}_{c,i(n)}\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{X}_{c(n)}^T \\
&\quad -\frac{1}{N_c}\mathbf{X}_{c(n)}\left(\mathbf{e}_{N_c}\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\mathbf{X}_{c,i(n)}^T \\
&\quad +\frac{1}{N_c^2}\mathbf{X}_{c(n)}\left(\mathbf{e}_{N_c}\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{X}_{c(n)}^T\Big]\mathbf{u}^{(n)} \\
&= \sum_{c=1}^{C}\mathbf{u}^{(n)^T}\mathbf{X}_{c(n)}\Big[\mathbf{U}_{\mathbf{\Pi}_{(n)}}^{\mathbf{I}_{N_c}}\cdot\mathbf{U}_{\mathbf{\Pi}_{(n)}^T}^{\mathbf{I}_{N_c}} \\
&\qquad -\frac{1}{N_c}\left(\mathbf{e}_{N_c}\otimes\mathbf{I}_{\psi(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\psi(n)}\right)\Big]\mathbf{X}_{c(n)}^T\mathbf{u}^{(n)} \\
&= \mathbf{u}^{(n)^T}\mathbf{X}_{(n)}\cdot diag\left(\mathbf{L}_1,\mathbf{L}_2,\cdots,\mathbf{L}_C\right)\cdot\mathbf{X}_{(n)}^T\mathbf{u}^{(n)}
\end{aligned}
\tag{A.2}
$$

where $\mathbf{L}_c$, $c=1,2,\cdots,C$, is defined as

$$
\mathbf{L}_c = \mathbf{U}_{\mathbf{\Phi}_{(n)}}^{\mathbf{I}_{N_c}}\mathbf{U}_{\mathbf{\Phi}_{(n)}^T}^{\mathbf{I}_{N_c}} - \frac{1}{N_c}\left(\mathbf{e}_{N_c}\otimes\mathbf{I}_{\Pi(n)}\right)\mathbf{u}_{\mathbf{\Phi}^{(n)}}\mathbf{u}_{\mathbf{\Phi}^{(n)}}^T\left(\mathbf{e}_{N_c}^T\otimes\mathbf{I}_{\Pi(n)}\right). \tag{A.3}
$$

A.2 Total-class Scatter Matrix of Tensor Samples

The total scatter matrix of tensor samples in Eq. (5.16) can be simplified as follows:

$$
\begin{aligned}
\mathbf{S}_t &= \sum_{m=1}^{M} \left(y_m - \bar{y}\right)\left(y_m - \bar{y}\right)^T \\
&= \sum_{m=1}^{M} \mathbf{u}^{(n)^T} \left(\mathbf{X}_{m(n)} - \overline{\mathbf{X}}_{(n)}\right) \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \left(\mathbf{X}_{m(n)} - \overline{\mathbf{X}}_{(n)}\right)^T \mathbf{u}^{(n)} \\
&= \sum_{m=1}^{M} \mathbf{u}^{(n)^T} \left[\mathbf{X}_{m(n)} - \mathbf{X}_{(n)} \left(\frac{1}{M}\mathbf{e}_M \otimes \mathbf{I}_{\Pi(n)}\right)\right] \mathbf{u}_{\mathbf{\Phi}(n)} \cdot \\
&\qquad\qquad \mathbf{u}_{\mathbf{\Phi}(n)}^T \left[\mathbf{X}_{m(n)}^T - \left(\frac{1}{M}\mathbf{e}_M^T \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{X}_{(n)}^T\right] \mathbf{u}_{(n)} \\
&= \sum_{m=1}^{M} \mathbf{u}^{(n)^T} \big[\mathbf{X}_{m(n)} \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \mathbf{X}_{m(n)}^T \\
&\qquad - \frac{1}{M}\mathbf{X}_{(n)} \left(\mathbf{e}_M \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \mathbf{X}_{m(n)}^T \\
&\qquad - \frac{1}{M}\mathbf{X}_{m(n)} \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \left(\mathbf{e}_M^T \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{X}_{(n)}^T \\
&\qquad + \frac{1}{M^2}\mathbf{X}_{(n)} \left(\mathbf{e}_M \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \left(\mathbf{e}_M^T \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{X}_{(n)}^T\Big] \mathbf{u}_{(n)} \\
&= \mathbf{u}^{(n)^T} \mathbf{X}_{(n)} \left[\mathbf{U}_{\mathbf{\Pi}_{(n)}}^{\mathbf{I}_{N_c}} \cdot \mathbf{U}_{\mathbf{\Pi}_{(n)}^T}^{\mathbf{I}_{N_c}} - \frac{1}{M}\left(\mathbf{e}_M \otimes \mathbf{I}_{\Pi(n)}\right) \mathbf{u}_{\mathbf{\Phi}(n)} \mathbf{u}_{\mathbf{\Phi}(n)}^T \left(\mathbf{e}_M^T \otimes \mathbf{I}_{\Pi(n)}\right)\right] \mathbf{X}_{(n)}^T \mathbf{u}^{(n)}
\end{aligned}
\tag{A.4}
$$

REFERENCES

[1] F. Mualla, S. Schöll, B. Sommerfeldt, A. Maier, and J. Hornegger, "Automatic cell detection in bright-field microscope images using SIFT, random forests, and hierarchical clustering," *IEEE Transactions on Medical Imaging*, vol. 32, no. 12, pp. 2274–2286, 2013.

[2] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M. A. Weber, T. Arbel, B. B. Avants, N. Ayache, P. Buendia, D. L. Collins, N. Cordier, J. J. Corso, A. Criminisi, T. Das, H. Delingette,. Demiralp, C. R. Durst, M. Dojat, S. Doyle, J. Festa, F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K. M. Iftekharuddin, R. Jena, N. M. John, E. Konukoglu, D. Lashkari, J. A. Mariz, R. Meier, S. Pereira, D. Precup, S. J. Price, T. R. Raviv, S. M. S. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H. C. Shin, J. Shotton, C. A. Silva, N. Sousa, N. K. Subbanna, G. Szekely, T. J. Taylor, O. M. Thomas, N. J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D. H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes, and K. V. Leemput, "The multimodal brain tumor image segmentation benchmark (BRATS)," *IEEE Transactions on Medical Imaging*, vol. 34, no. 10, pp. 1993–2024, 2015.

[3] G. AlRegib, M. Deriche, Z. Long, H. Di, Z. Wang, Y. Alaudah, M. A. Shafiq, and M. Alfarraj, "Subsurface structure analysis using computational interpretation and learning: A visual signal processing perspective," *IEEE Signal Processing Magazine*, vol. 35, no. 2, pp. 82–98, 2018.

[4] M. Z. Lukawski, B. J. Anderson, C. Augustine, L. E. Capuano, K. F. Beckers, B. Livesay, and J. W. Tester, "Cost analysis of oil, gas, and geothermal well drilling," *Journal of Petroleum Science and Engineering*, vol. 118, pp. 1–14, 2014.

[5] Ö. Yilmaz, *Seismic data analysis*. 2001, vol. 1.

[6] L. J. Griffiths, F. R. Smolka, and L. D. Trembly, "Adaptive deconvolution: A new technique for processing time-varying seismic data," *Geophysics*, vol. 42, no. 4, pp. 742–759, 1977.

[7] G. F. Margrave, M. P. Lamoureux, and D. C. Henley, "Gabor deconvolution: Estimating reflectivity by nonstationary deconvolution of seismic data," *Geophysics*, vol. 76, no. 3, W15–W30, 2011.

[8] T. Alkhalifah and I. Tsvankin, "Velocity analysis for transversely isotropic media," *Geophysics*, vol. 60, no. 5, pp. 1550–1566, 1995.

[9] G. Liu, S. Fomel, L. Jin, and X. Chen, "Stacking seismic data using local correlation," *Geophysics*, vol. 74, no. 3, pp. V43–V48, 2009.

[10] S. H. Gray, J. Etgen, J. Dellinger, and D. Whitmore, "Seismic migration problems and solutions," *Geophysics*, vol. 66, no. 5, pp. 1622–1640, 2001.

[11] P. Sava and S. Fomel, "Time-shift imaging condition in seismic migration," *Geophysics*, vol. 71, no. 6, S209–S217, 2006.

[12] H Bouzas, *The challenges of oil and gas data interpretation*, `https://bldl.ii.uib.no/bldl-opening-presentations/Bouzas-BergenV1-Bouzas.pdf`.

[13] dGB Earth Sciences, *OpendTect*, `http://www.opendtect.org/`.

[14] Schlumberger, *Petrel E&P Software Platform*, `https://www.software.slb.com/products/petrel`.

[15] M. T. Taner, F. Koehler, and R. E. Sheriff, "Complex seismic trace analysis," *Geophysics*, vol. 44, no. 6, pp. 1041–1063, 1979.

[16] S. Sinha, P. S. Routh, P. D. Anno, and J. P. Castagna, "Spectral decomposition of seismic data with continuous-wavelet transform," *Geophysics*, vol. 70, no. 6, P19–P25, 2005.

[17] R. Neelamani and D. R. Converse, *Geologic features from curvelet based seismic attributes*, US Patent 8,538,702, 2013.

[18] O. Rioul and M. Vetterli, "Wavelets and signal processing," *IEEE Signal Processing Magazine*, vol. 8, no. 4, pp. 14–38, 1991.

[19] J.-L. Starck, E. J. Candès, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions on Image Processing*, vol. 11, no. 6, pp. 670–684, 2002.

[20] I. Sobel, "An isotropic 3× 3 image gradient operator," *Machine Vision for Three-dimensional Scenes*, pp. 376–379, 1990.

[21] A. Amin and M. Deriche, "A new approach for salt dome detection using a 3D multidirectional edge detector," *Applied Geophysics*, vol. 12, no. 3, pp. 334–342, 2015.

[22] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[23] A. Amin, M. Deriche, M. A. Shafiq, Z. Wang, and G. AlRegib, "Automated salt-dome detection using an attribute ranking framework with a dictionary-based classifier," *Interpretation*, vol. 5, no. 3, SJ61–SJ79, 2017.

[24] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, 1972.

[25] P. Jacquemin and J. Mallet, "Automatic faults extraction using double Hough transform," in *SEG Technical Program Expanded Abstracts 2005*, 2005, pp. 755–758.

[26] N. M. AlBinHassan and K. J. Marfurt, "Fault detection using Hough transforms," in *SEG Technical Program Expanded Abstracts 2003*, 2005, pp. 1719–1721.

[27] M. A. Shafiq, T. Alshawi, Z. Long, and G. AlRegib, "The role of visual saliency in the automation of seismic interpretation," *Geophysical Prospecting*, vol. 66, no. S1, pp. 132–143, 2018.

[28] Z. Wang, D. Temel, and G. AlRegib, "Fault detection using color blending and color transformations," in *Proceedings of 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 999–1003.

[29] H. Di and G. AlRegib, "Seismic multi-attribute classification for salt boundary detection: A comparison," in *79th EAGE Conference and Exhibition*, 2017.

[30] A. Berthelot, A. H. Solberg, and L.-J. Gelius, "Texture attributes for detection of salt," *Journal of Applied Geophysics*, vol. 88, pp. 52–69, 2013.

[31] Z. Zheng, P. Kavousi, and H. Di, "Multi-attributes and neural network-based fault detection in 3D seismic interpretation," in *Civil, Structural and Environmental Engineering*, ser. Advanced Materials Research, vol. 838, 2014, pp. 1497–1502.

[32] H. Di, Z. Wang, and G. AlRegib, "Seismic fault detection from post-stack amplitude by convolutional neural networks," in *80th EAGE Conference and Exhibition*, 2018.

[33] H. Di, Z. Wang, and G. AlRegib, "Deep convolutional neural networks for seismic salt-body delineation," in *AAPG Annual Convention and Exhibition*, 2018.

[34] R. E. Sheriff, "Factors affecting seismic amplitudes," *Geophysical Prospecting*, vol. 23, no. 1, pp. 125–138, 1975.

[35] J. H. Bodine, "Waveform analysis with seismic attributes," in *SEG Technical Program Expanded Abstracts 1984*, 1984, pp. 505–509.

[36] R. E. White, "Properties of instantaneous seismic attributes," *The Leading Edge*, vol. 10, no. 7, pp. 26–32, 1991.

[37] Y. Luo, W. G. Higgs, and W. S. Kowalik, "Edge detection and stratigraphic analysis using 3D seismic data," in *SEG Technical Program Expanded Abstracts 1996*, 1996, pp. 324–327.

[38] A. E. Barnes, "Theory of two-dimensional complex seismic trace analysis," *Geophysics*, vol. 61, no. 1, pp. 264–272, 1996.

[39] M. S. Bahorich and S. L. Farmer, "3-D seismic discontinuity for faults and stratigraphic features: The coherence cube," *The Leading Edge*, vol. 14, no. 10, pp. 1053–1058, 1995.

[40] J. P. Castagna, S. Sun, and R. W. Siegfried, "Instantaneous spectral analysis: Detection of low-frequency shadows associated with hydrocarbons," *The Leading Edge*, vol. 22, no. 2, pp. 120–127, 2003.

[41] I. Daubechies, J. Lu, and H. T. Wu, "Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 243–261, 2011.

[42] Y. Chen, T. Liu, X. Chen, J. Li, and E. Wang, "Time-frequency analysis of seismic data using synchrosqueezing wavelet transform," *Journal of Seismic Exploration*, vol. 23, no. 5, pp. 303–312, 2014.

[43] J. P. Amezquita-Sanchez and H. Adeli, "A new music-empirical wavelet transform methodology for time–frequency analysis of noisy nonlinear and non-stationary signals," *Digital Signal Processing*, vol. 45, pp. 55–68, 2015.

[44] K. J. Marfurt, V. Sudhaker, A. Gersztenkorn, K. D. Crawford, and S. E. Nissen, "Coherency calculations in the presence of structural dip," *Geophysics*, vol. 64, no. 1, pp. 104–111, 1999.

[45] M. A. Shafiq, Z. Wang, A. Amin, T. Hegazy, M. Deriche, and G. AlRegib, "Detection of salt-dome boundary surfaces in migrated seismic volumes using gradient of textures," in *SEG Technical Program Expanded Abstracts 2015*, 2015, pp. 1811–1815.

[46] M. A. Shafiq and G. AlRegib, "Perceptual and non-perceptual dissimilarity measures for salt dome delineation," in *78th EAGE Conference and Exhibition*, 2016.

[47] Z. Wang, Z. Long, and G. AlRegib, "Tensor-based subspace learning for tracking salt-dome boundaries constrained by seismic attributes," in *Proceedings of*

2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 1125–1129.

[48] P. Bakker, L. J. Van Vliet, and P. W. Verbeek, "Edge preserving orientation adaptive filtering," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999, pp. 535–540.

[49] C. Höcker and G. Fehmers, "Fast structural interpretation with structure-oriented filtering," *The Leading Edge*, vol. 21, no. 3, pp. 238–243, 2002.

[50] *Fault in road cut near Kingman, Arizona*, `http://epod.usra.edu/blog/2016/12/fault-in-road-cut-near-kingman-arizona.html`, Dec. 2016 (accessed Mar. 2018).

[51] *Deep water reservoir*, `https://petroleumstudies.wordpress.com/2008/11/17/`, Nov. 2008 (accessed Mar. 2018).

[52] P. P. Van Bemmel and R. E. F. Pepper, *Seismic signal processing method and apparatus for generating a cube of variance values*, US Patent 6,151,555, 2000.

[53] A. Roberts, "Curvature attributes and their application to 3D interpreted horizons," *First Break*, vol. 19, no. 2, pp. 85–100, 2001.

[54] K. M. Tingdahl and M. De Rooij, "Semi-automatic detection of faults in 3D seismic data," *Geophysical Prospecting*, vol. 53, no. 4, pp. 533–542, 2005.

[55] I. Cohen, N. Coult, and A. A. Vassiliou, "Detection and extraction of fault surfaces in 3D seismic data," *Geophysics*, vol. 71, no. 4, P21–P27, 2006.

[56] H. Di and D. Gao, "3D seismic flexure analysis for subsurface fault detection and fracture characterization," *Pure and Applied Geophysics*, vol. 174, no. 3, pp. 747–761, 2017.

[57] K. J. Marfurt, R. L. Kirlin, S. L. Farmer, and M. S. Bahorich, "3-D seismic attributes using a semblance-based coherency algorithm," *Geophysics*, vol. 63, no. 4, pp. 1150–1165, 1998.

[58] A. Gersztenkorn and K. J. Marfurt, "Eigenstructure-based coherence computations as an aid to 3-D structural and stratigraphic mapping," *Geophysics*, vol. 64, no. 5, pp. 1468–1479, 1999.

[59] T. Yang, B. Zhang, and J. Gao, "A fast algorithm for coherency estimation in seismic data based on information divergence," *Journal of Applied Geophysics*, vol. 115, pp. 140–144, 2015.

[60] F. Li and W. Lu, "Coherence attribute at different spectral scales," *Interpretation*, vol. 2, no. 1, SA99–SA106, 2014.

[61] J. Sui, X. Zheng, and Y. Li, "A seismic coherency method using spectral amplitudes," *Applied Geophysics*, vol. 12, no. 3, pp. 353–361, 2015.

[62] Y. Alaudah and G. AlRegib, "A generalized tensor-based coherence attribute," in *78th EAGE Conference and Exhibition*, 2016.

[63] ——, "A directional coherence attribute for seismic interpretation," in *SEG Technical Program Expanded Abstracts 2017*. 2017, pp. 2102–2106.

[64] Z. Wang and G. AlRegib, "Fault detection in seismic datasets using Hough transform," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 2372–2376.

[65] G. C. Fehmers and C. F. Höcker, "Fast structural interpretation with structure-oriented filtering," *Geophysics*, vol. 68, no. 4, pp. 1286–1293, 2003.

[66] D. Hale, "Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images," *Geophysics*, vol. 78, no. 2, O33–O43, 2013.

[67] S. I. Pedersen, T. Randen, S. L., and Ø. Steen, "Automatic fault extraction using artificial ants," in *SEG Technical Program Expanded Abstracts 2002*, 2002, pp. 512–515.

[68] C. C. Silva, C. S. Marcolino, and F. D. Lima, "Automatic fault extraction using ant tracking algorithm in the Marlim South Field, Campos Basin," in *SEG Technical Program Expanded Abstracts 2005*. 2005, pp. 857–860.

[69] T. Randen, E. Monsen, C. Signer, A. Abrahamsen, J. O. Hansen, T. Sæter, J. Schlaf, and L. Sønneland, "Three-dimensional texture attributes for seismic data analysis," in *SEG Technical Program Expanded Abstracts 2000*, 2000, pp. 668–671.

[70] N. M. AlBinHassan, Y. Luo, and M. N. Al-Faraj, "3D edge-preserving smoothing and applications," *Geophysics*, vol. 71, no. 4, P5–P11, 2006.

[71] T. Pavlidis, "A thinning algorithm for discrete binary images," *Computer Graphics and Image Processing*, vol. 13, no. 2, pp. 142–157, 1980.

[72] D. Gibson, M. Spann, J. Turner, and T. Wright, "Fault surface detection in 3D seismic data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 9, pp. 2094–2102, 2005.

[73] Z. Wang and G. AlRegib, "Automatic fault surface detection by using 3D Hough transform," in *SEG Technical Program Expanded Abstracts 2014*, 2014, pp. 1439–1444.

[74] X. Wu and D. Hale, "3D seismic image processing for faults," *Geophysics*, vol. 81, no. 2, pp. IM1–IM11, 2016.

[75] D. Hale and J. Emanuel, "Seismic interpretation using global image segmentation," in *Expanded Abstracts of the SEG 73rd Annual Meeting*, 2003, pp. 2410–2413.

[76] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[77] Y. Gdalyahu, D. Weinshall, and M. Werman, "Self-organization in vision: Stochastic clustering for image segmentation, perceptual grouping, and image database organization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 10, pp. 1053–1074, 2001.

[78] B. Zhang, Y. Liu, P. M., and H. N., "Semi-automated fault interpretation based on seismic attributes," in *Expanded Abstracts of the SEG 83rd Annual Meeting*, 2013, pp. 1498–1503.

[79] H. Di, D. Gao, and D. Martion, "Seismic attribute-aided fault detection in petroleum industry: A review," *Fault detection: Methods, Applications and Technology: Nova Science Publishers*, pp. 53–80, 2017.

[80] Z. Zheng, P. Kavousi, and H. Di, "Multi-attributes and neural network-based fault detection in 3D seismic interpretation," in *Advanced Materials Research*, vol. 838, 2014, pp. 1497–1502.

[81] L. Huang, X. Dong, and T. E. Clee, "A scalable deep learning platform for identifying geologic features from seismic attributes," *The Leading Edge*, vol. 36, no. 3, pp. 249–256, 2017.

[82] H. Di, M. A. Shafiq, and G. AlRegib, "Seismic-fault detection based on multi-attribute support vector machine analysis," in *SEG Technical Program Expanded Abstracts 2017*. 2017, pp. 2039–2044.

[83] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[84] T. Zhao, V. Jayaram, A. Roy, and K. J. Marfurt, "A comparison of classification techniques for seismic facies recognition," *Interpretation*, vol. 3, no. 4, SAE29–SAE58, 2015.

[85] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. 2012, pp. 1097–1105.

[86] H. Di, Z. Wang, and G. AlRegib, "Seismic fault detection from post-stack amplitude by convolutional neural networks," in *80th EAGE Conference and Exhibition*, 2018.

[87] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[88] *Jashak salt dome, the sweet salinity*, `http://www.iranreview.org/content/Documents/Jashak-Salt-Dome-the-Sweet-Salinity.htm`, Apr. 2017 (accessed Apr. 2017).

[89] *Structural trap*, `https://en.wikipedia.org/wiki/Structural_trap`, Feb. 2017 (accessed Apr. 2017).

[90] T. Hegazy and G. AlRegib, "Texture attributes for detecting salt bodies in seismic data," in *Expanded Abstracts of the SEG 84th Annual Meeting*, 2014, pp. 1455–1459.

[91] S. Chopra and K. J. Marfurt, "Understanding the seismic disorder attribute and its applications," *The Leading Edge*, vol. 35, no. 8, pp. 695–702, 2016.

[92] X. Wu, "Methods to compute salt likelihoods and extract salt boundaries from 3d seismic images," *Geophysics*, vol. 81, no. 6, pp. IM119–IM126, 2016.

[93] J. Lomask, B. Biondi, and J. Shragge, "Image segmentation for tracking salt boundaries," in *Expanded Abstracts of the SEG 74th Annual Meeting*, 2004, pp. 2443–2446.

[94] J. Lomask, R. G. Clapp, and B. Biondi, "Application of image segmentation to tracking 3D salt boundaries," *Geophysics*, vol. 72, no. 4, P47–P56, 2007.

[95] A. D. Halpert, R. G. Clapp, and B. Biondi, "Seismic image segmentation with multiple attributes," in *Expanded Abstracts of the SEG 79th Annual Meeting*, 2009, pp. 3700–3704.

[96] ——, "Speeding up seismic image segmentation," in *Expanded Abstracts of the SEG 80th Annual Meeting*, 2010, pp. 1276–1280.

[97] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[98] Z. J., Z. Y., C. Z., and L. J., "Detecting boundary of salt dome in seismic data with edge detection technique," in *Society of Exploration Geophysicists*, 2007.

[99] A. A. Aqrawi, T. H. Boe, and S. Barros, "Detecting salt domes using a dip guided 3D Sobel seismic attribute," in *Expanded Abstracts of the SEG 81st Annual Meeting*, 2011, pp. 1014–1018.

[100] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[101] S. Chopra and K. J. Marfurt, *Seismic Attributes for Prospect Identification and Reservoir Characterization*. 2007.

[102] A. Amin, M. Deriche, T. Hegazy, Z. Wang, and G. AlRegib, "A novel approach for salt dome detection using a dictionary-based classifier," in *Expanded Abstracts of the SEG 85th Annual Meeting*, 2015, pp. 1816–1820.

[103] Z. Wang, T. Hegazy, Z. Long, and G. AlRegib, "Noise-robust detection and tracking of salt domes in postmigrated volumes using texture, tensors, and subspace learning," *Geophysics*, vol. 80, no. 6, WD101–WD116, 2015.

[104] P. Guillen, G. Larrazabal, G. González, D. Boumber, and R. Vilalta, "Supervised learning to detect salt body," in *SEG Technical Program Expanded Abstracts 2015*, 2015, pp. 1826–1829.

[105] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer, 2009.

[106] E. Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classification," *Biometrics*, vol. 21, no. 3, pp. 768–769, 1965.

[107] H. Di, M. Shafiq, and G. AlRegib, "Multi-attribute k-means cluster analysis for salt boundary detection," in *79th EAGE Conference and Exhibition*, 2017.

[108] H. Di and G. AlRegib, "Seismic multi-attribute classification for salt boundary detection - a comparison," in. 2017.

[109] M. A. Shafiq, Z. Wang, G. AlRegib, A. Amin, and M. Deriche, "A texture-based interpretation workflow with application to delineating salt domes," *Interpretation*, vol. 5, no. 3, SJ1–SJ19, 2017.

[110] J. Canny, "A computational approach to edge detection," in *Readings in Computer Vision*, 1987, pp. 184–203.

[111] H. Di, Z. Wang, and G. AlRegib, "Deep convolutional neural networks for seismic salt-body delineation," in *Abstracts of American Association of Petroleum Geologists (AAPG)*. 2017.

[112] G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, and D. Kriegman, "Parameterized families of polynomials for bounded algebraic curve and surface fitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 3, pp. 287–303, 1994.

[113] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, "The 3D Hough transform for plane detection in point clouds: A review and a new accumulator design," *3D Research*, vol. 2, no. 2, pp. 1–13, 2011.

[114] H. Alt and M. Godau, "Computing the fréchet distance between two polygonal curves," *International Journal of Computational Geometry & Applications*, vol. 5, pp. 75–91, 1995.

[115] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 850–863, 1993.

[116] dGB Earth Sciences, *Netherlands offshore F3 block complete 4 GB*, `http://opendtect.org/osr/pmwiki.php/Main/NetherlandsOffshoreF3BlockCompl`

[117] L. De Lathauwer and B. De Moor, "From matrix to tensor: Multilinear algebra and signal processing," in *Institute of Mathematics and Its Applications Conference Series*, vol. 67, 1998, pp. 1–16.

[118] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-$(R_1,R_2,\cdots,R_n)$ approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.

[119] P. A. Regalia and M. K. Sanjit, "Kronecker products, unitary matrices, and signal processing applications," *SIAM Review*, vol. 31, no. 4, pp. 586–613, 1989.

[120] T. Hegazy and G. AlRegib, "COHERENSi: A new full-reference IQA index using error spectrum chaos," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 965–969.

[121] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[122] D. A. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008.

[123] R. E. Roger, "A faster way to compute the noise-adjusted principal components transform matrix," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 6, pp. 1194–1196, 1994.

[124] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings, "A new autocovariance least-squares method for estimating noise covariances," *Automatica*, vol. 42, no. 2, pp. 303–308, 2006.

[125] T. Y. Kong and A. Rosenfeld, Eds., *Topological algorithms for digital image processing*. Elsevier Science Inc., 1996.

[126] J. Toriwaki and H. Yoshida, *Fundamentals of three-dimensional digital image processing*. Springer Science & Business Media, 2009.

[127] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Sixth International Conference on Computer Vision*, 1998, pp. 839–846.

[128] S. Chopra and K. J. Marfurt, *Causes and appearance of noise in seismic data volumes*, `http://www.aapg.org/publications/news/explorer/column/articleid/12428/causes-and-appearance-of-noise-in-seismic-data-volumes#prettyPhoto`, 2014.

[129] D. Cai, X. He, J. Han, and H.-J. Zhang, "Orthogonal Laplacianfaces for face recognition," *IEEE transactions on Image Processing*, vol. 15, no. 11, pp. 3608–3614, 2006.

[130] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "MPCA: Multilinear principal component analysis of tensor objects," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 18–39, 2008.

[131] Z. Lai, W. K. Wong, Y. Xu, C. Zhao, and M. Sun, "Sparse alignment for robust tensor learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 10, pp. 1779–1792, 2014.

VITA

Zhen Wang received his bachelor's degree in information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2010. In 2012, he received his master's degrees in information and communication engineering from Shanghai Jiao Tong University and electrical and computer engineering (ECE) from Georgia Tech, Atlanta, GA, USA. He is currently a graduate research assistant working on his doctoral degree in ECE at Georgia Tech. As a Ph.D. student of Dr. Ghassan AlRegib, he is an active member in both the Omni Lab for Intelligent Visual Engineering and Science (OLIVES) and the Center for Energy and Geo Processing (CEGP). His research interests lie in image and video processing, machine learning, computer vision, and seismic interpretation.